NAMA:

NUR HASINAH BINTI HASSAN

NOMBOR MATRIK:

WEK000295

TOOLS FOR JAWI

CHARACTER RECOGNITION

NAMA PENASIHAT:

ENCIK ZAIDI RAZAK

NAMA MODERATOR:

ENCIK MOHD YAMANI IDNA BIN IDRIS

ABSTRAK

Huruf jawi terdiri daripada 36 huruf. Ia merupakan huruf-huruf yang kebanyakannya lahir daripada huruf arab.Tulisannya adalah unik kerana mempunyai aksara yang bersambung dalam sesuatu perkataan. Ini menyebabkan pengecaman terhadap tulisan jawi tangan menjadi lebih kompleks apabila aksara perlu dipecahkan. Selain itu, perwakilan bagi sesuatu aksara jawi mempunyai bentuk yang berlainan bergantung kepada kedudukannya dalam perkataan.

Perisian alatan teknikal iaitu MATLAB digunakan untuk membina alatan yang terdiri daripada pemprosesan imej, pensegmenan, dan pengecaman aksara jawi. Input adalah satu perkataan jawi yang diimbas. Imej dalam bentuk digital akan melalui pemprosesan imej untuk tujuan pengurangan kebisingan dan peningkatan kualiti.

Imej seterusnya dipecahkan kepada aksara-aksara tunggal dalam proses pensegmenan. Pengecaman terhadap aksara-aksara tersebut akan dilaksanakan menerusi pembacaan kod rantaian menggunakan kaedah lapan arah. Sebuah pangkalan pengetahuan akan dibina untuk menyimpan kesemua kod rantaian bagi mewakili setiap aksara. Proses pemadanan akan dilakukan untuk membandingkan kod rantaian yang diperolehi dengan kod rantaian yang disimpan dalam pangkalan pengetahuan.

Output yang akan didapati ialah aksara-aksara jawi tunggal yang mewakili perkataan yang diinput.

PENGHARGAAN

Syukur saya ke hadrat Ilahi kerana akhirnya laporan Projek Ilmiah Tahap Akhir ini dapat disiapkan dengan jayanya. Saya ingin merakamkan setinggi-tinggi penghargaan kepada Encik Zaidi Razak yang berperanan sebagai penasihat dalam kajian ini. Beliau telah banyak memberikan tunjuk ajar dan sokongan yang sepenuhnya untuk memastikan kajian ini dapat dilaksanakan.

Sekalung penghargaan kepada moderator saya iaitu Encik Mohd Yamani Idna b. Idris. Beliau banyak membantu dalam memberikan pendapat dan berkongsi pandangan dalam merealisasikan projek yang telah dirancang.

Saya ingin merakamkan jutaan terima kasih kepada pihak Fakulti Sains Komputer dan Teknologi Maklumat yang bertanggungjawab menyediakan kemudahan perisian yang diperlukan dan kemudahan makmal untuk membangunkan projek ini..

Terima kasih juga kepada kedua ibu bapa saya yang banyak memberikan dorongan dan galakan selama ini.

Seterusnya, tidak lupa kepada rakan-rakan yang sama-sama berusaha secara bersungguh-sungguh untuk menyiapkan projek ini.

Semoga hasil kajian yang didapati dapat dimanfaatkan sepenuhnya dan memberi impak positif kepada perkembangan kajian tulisan jawi.

Sekian, terima kasih.

SENARAI ISI KANDUNGAN

SENARAI ISI KANDUNGAN

SENARAI RAJAH

SENARAI RAJAH

SENARAI JADUAL

1. PENGENALAN

Kajian yang dilakukan ialah membina alatan yang lengkap untuk tujuan pengecaman tulisan tangan jawi. Alatan yang dibangunkan terdiri daripada proses peningkatan kualiti imej (*image quality enhancement*), pengurangan gangguan (*noise reduction*), pensegmenan (*segmentation*), dan pengecaman (*recognition*).

Pengecaman tulisan jawi memerlukan imej berbentuk digital. Maka imej yang diimbas akan ditukar kepada imej digital dan disimpan ke dalam fail. Pemprosesan imej akan dilakukan ke atas imej tersebut. Ia bertujuan untuk menghasilkan imej yang berkualiti bagi memudahkan proses seterusnya dilaksanakan.

Proses segmentasi adalah proses yang digunakan untuk mengasingkan aksara jawi yang bersambung dan tidak bersambung. Kemudian proses pengecaman dilakukan untuk mengenalpasti huruf jawi tersebut. Sebuah pangkalan pengetahuan akan dibangunkan untuk tujuan pengecaman. Hasil yang akan diperolehi ialah aksara jawi tunggal yang mewakili perkataan jawi yang diinput.

OBJEKTIF

Terdapat beberapa objektif yang telah ditetapkan untuk dicapai. Antara objektif tersebut adalah seperti berikut:

1. Mengecam imej tulisan tangan jawi dengan memaparkan sebagai aksara jawi tunggal.
2. Mendigitalkan imej tulisan jawi yang diinput dan seterusnya meningkatkan kualiti imej dan mengurangkan gangguan imej tersebut.
3. Melakukan proses pensegmenan dan pemadanan untuk tujuan pengecaman aksara.
4. Membangunkan pangkalan pengetahuan untuk menyimpan kod rantaian (*chain code*) dan imej digital bagi aksara tertentu.
5. Mengatasi masalah kekurangan penterjemah mahir dalam penulisan jawi yang mempunyai gaya penulisan yang tersendiri.
6. Memudahkan penterjemahan kitab-kitab atau buku-buku lama yang mempunyai tulisan yang kabur atau tidak jelas.

SKOP PROJEK

Pengecaman tulisan jawi mempunyai beberapa skop yang telah ditetapkan. Ini bertujuan bagi menghadkan kajian yang dilakukan supaya tumpuan terhadap masalah yang perlu diselesaikan dapat diberikan secara khusus.

Pengecaman hanya melibatkan satu patah perkataan jawi yang diinput sahaja. Ini bermakna kajian yang dilakukan bermula dengan satu patah perkataan dan akan diikuti dengan satu ayat sekiranya pengecaman berjaya dilakukan.

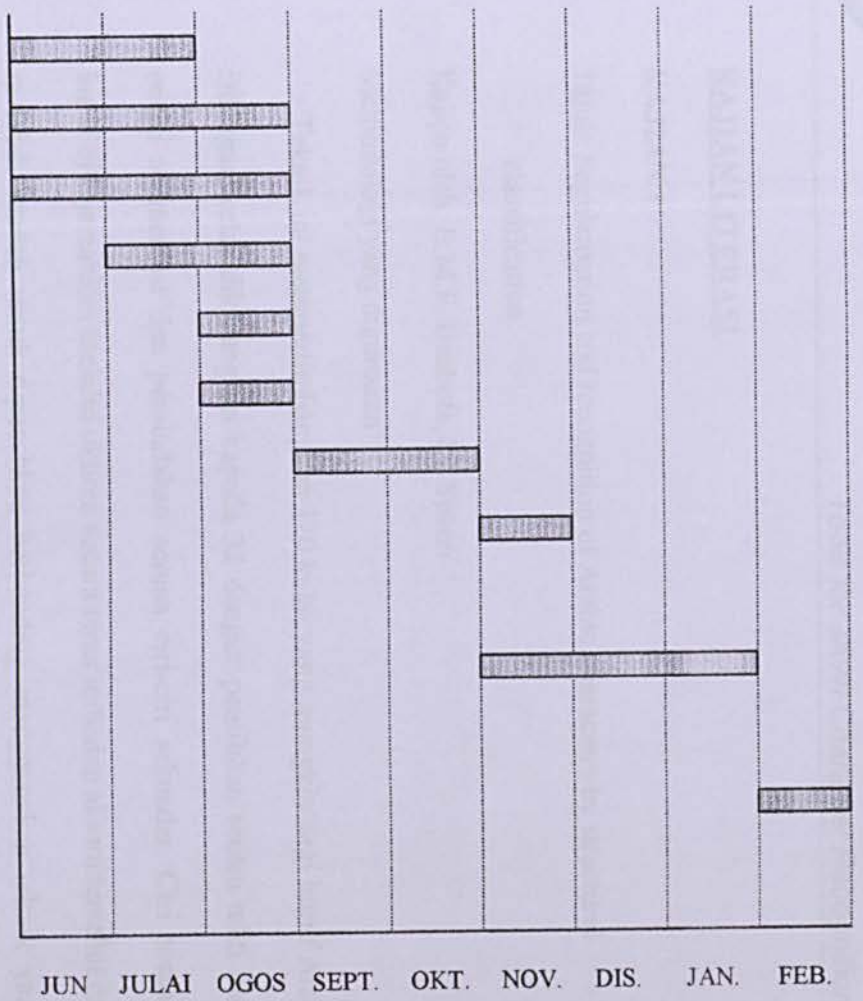
Seterusnya pengecaman tidak dilakukan secara atas talian (*off-line recognition.*). Tulisan jawi akan dibaca dan diinput melalui alat pengimbas setelah perkataan jawi tersebut lengkap ditulis. Berbeza dengan pengecaman atas talian di mana pengecaman dilakukan semasa aksara diinput menggunakan peranti tertentu yang diperlukan untuk menulis.

Pemadanan yang dilakukan adalah bergantung kepada pangkalan pengetahuan yang mempunyai kod rantaian aksara tertentu. Ini termasuk kedudukan yang berlainan sama ada di hadapan, di tengah, di hujung atau aksara tunggal bagi mewakili sesuatu huruf jawi yang sama.

PERANCANGAN PERLAKSAANAAN PROJEK

AKTIVITI:

1. Kajian dan pemahaman terhadap tajuk projek
2. Mempelajari perisian Matlab yang akan digunakan.
3. Memahami konsep pemprosesan imej secara teori.
4. Kajian terhadap methodologi yang digunakan.
5. Membuat spesifikasi proses rekabentuk
6. Merekabentuk pangkalan pengetahuan
7. Pemprosesan terhadap imej digital.
8. Membangunkan pangkalan pengetahuan dengan memasukkan aksara yang dipilih.
9. Membina aturcara program untuk proses pengecaman.
10. Membuat ujian terhadap sistem yang telah dibina.



2. KAJIAN LITERASI

KAJIAN 1

Tajuk: Segmentation and recognition of Arabic characters by structural classification

Kajian oleh: B.M.F. Bushofa, M. Spann

Methodologi yang digunakan:

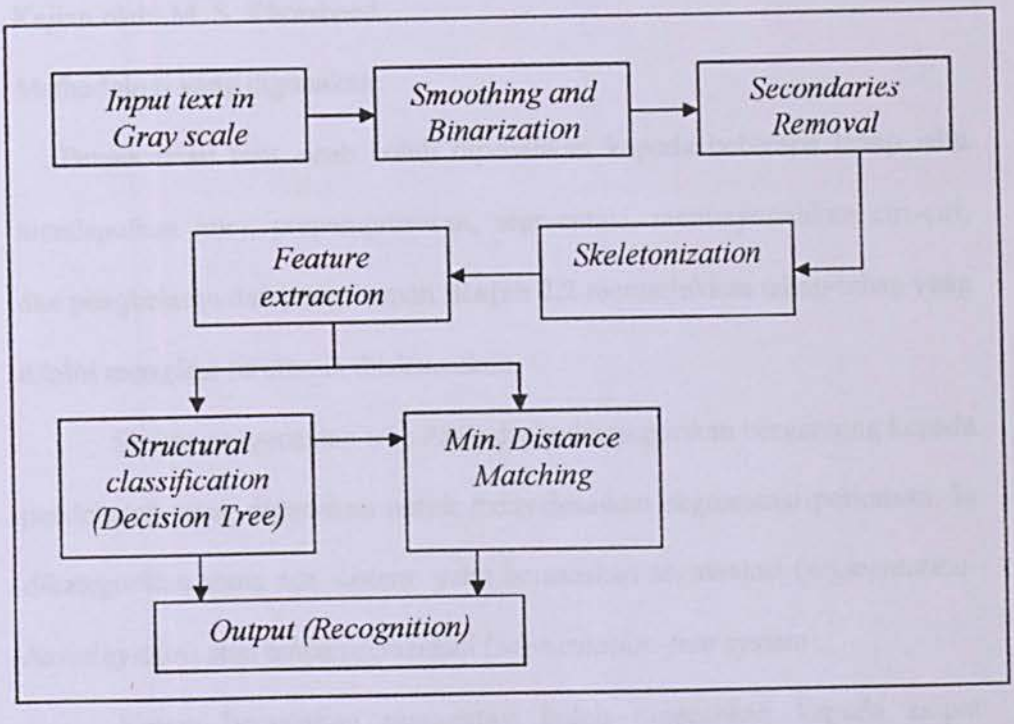
Teknik ini mempertimbangkan 120 kelas yang mungkin bagi huruf Arab. Bilangan kelas dikurangkan kepada 32 dengan pemilihan secara teliti oleh posisi segmentasi dan pemindahan semua ciri-ciri sekunder. Ciri sesuatu huruf diterjemahkan melalui ukuran secara terus terhadap aksara tersebut dan ia lebih mudah untuk dipecahkan berbanding menggunakan teknik yang berasaskan gambaran *Fourier*.

Ciri ini terdiri daripada segmentasi garisan dan lengkok yang disesuaikan dengan poligon menggunakan metod yang baru dibangunkan. Kedua-dua ciri segmentasi dan maklumat tentang ciri sekunder digunakan untuk mengkelaskan aksara kepada dua tahap. Pada tahap pertama, pepohon keputusan digunakan untuk mengkelaskan aksara menggunakan ciri yang minima. Sekiranya mana-mana aksara dibuang pada tahap ini, ia akan dihantar ke tahap kedua dan dibandingkan dengan set bagi templet yang ditakrifkan.

Proses segmentasi memecahkan perkataan kepada huruf. Oleh sebab perkataan Arab pada kebiasaanya bersambung pada garis asas, kebanyakan algoritma mempergunakan fakta ini. Dalam algoritma segmentasi yang direkabentuk, ia adalah mustahil untuk menentukan kelas yang betul.

Salah satu cara yang digunakan ialah algoritma yang memeriksa adanya sudut atau pepenjuru yang ditunjukkan oleh dua huruf yang bersambung. Pepenjuru ini terjadi pada garis asas yang terdiri daripada bilangan maksimum bagi piksel hitam.

Apabila baris asas dijumpai, algoritma diteruskan dengan membuat imbasan terhadap imej dari kanan ke kiri di atas garisan asas. Ia menggunakan tetingkap 7x7 dan memeriksa kejiranan bagi piksel yang berada ditengah-tengah. **Rajah 2.1** menunjukkan gambarajah blok bagi sistem ini.



Rajah 2.1: Gambarajah blok sistem.

Kesimpulan:

Teknik ini mengurangkan bilangan kelas huruf Arab daripada 120 kepada 32 dengan kaedah segmentasi dan memindahkan ciri-ciri sekunder. *Stroke* utama diskeletonkan dan ciri-cirinya diterjemahkan dengan pengukuran secara terus terhadap aksara tersebut.

KAJIAN 2

Tajuk: Off-Line Arabic Character Recognition – A Review

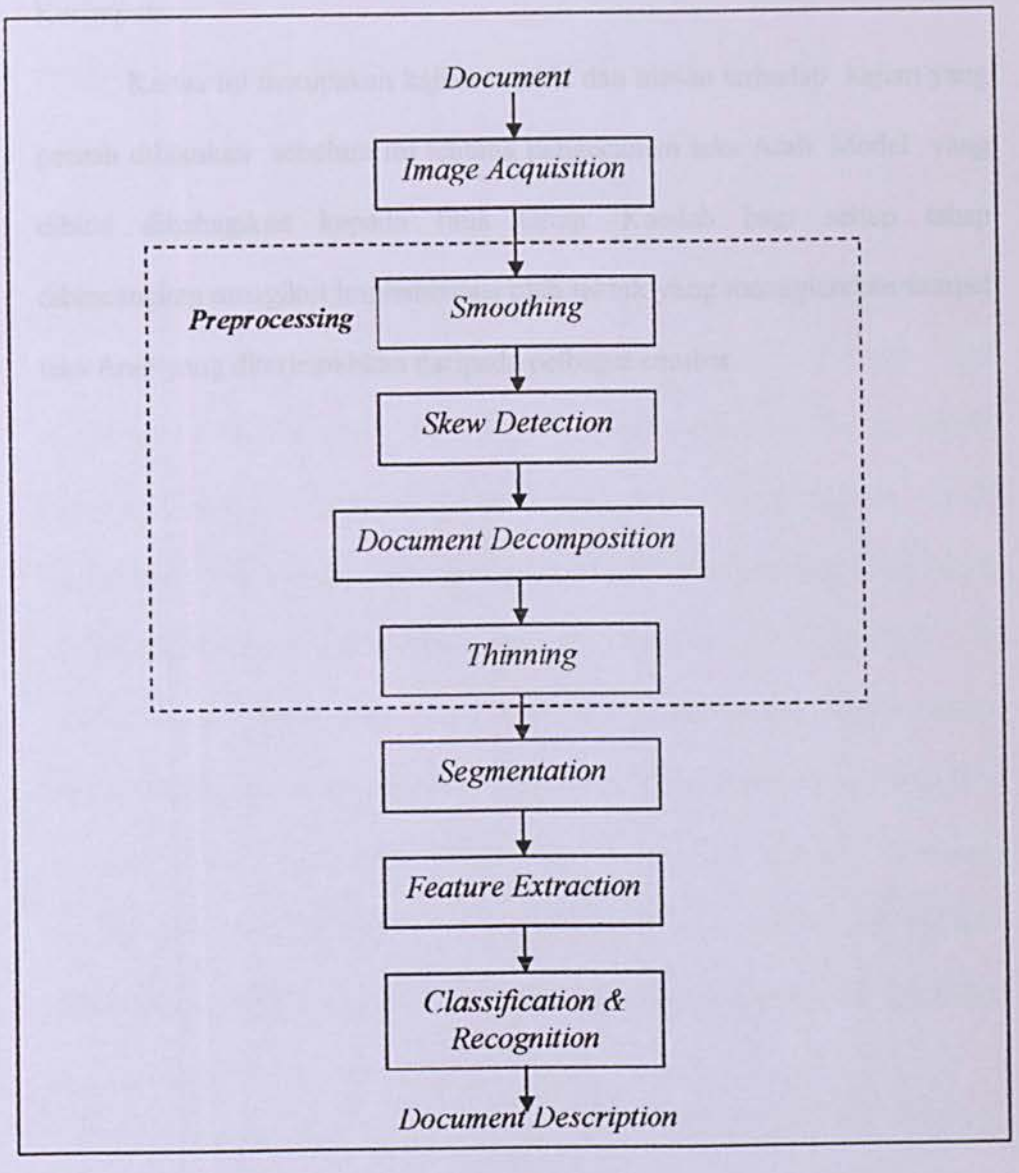
Kajian oleh: M. S. Khorsheed

Methodologi yang digunakan:

Pengecaman teks Arab boleh dipecahkan kepada beberapa tahap iaitu mendapatkan imej, prapemprosesan, segmentasi, menterjemahkan ciri-ciri, dan pengkelasan dan pengecaman. **Rajah 2.2** menunjukkan tahap-tahap yang dilalui mengikut turutan ia dilaksanakan.

Sistem pengecaman teks Arab dapat dikategorikan bergantung kepada pendekatan yang digunakan untuk menyelesaikan segmentasi perkataan. Ia dikategorikan sama ada sistem yang berasaskan segmentasi (*segmentation-based* system) atau tanpa segmentasi (*segmentation-free* system).

Sistem berasaskan segmentasi boleh dipecahkan kepada empat kategori. Kategori pertama ialah aksara yang terasing atau prasegmen. Kategori kedua ialah pensegmenan perkataan kepada aksara. Pensegmenan perkataan kepada ciri-ciri primitif merupakan kategori ketiga dan akhir sekali ialah integrasi bagi pengecaman dan pensegmenan.



Rajah 2.2: Diagram untuk sistem pengecaman teks Arab

Kesimpulan:

Kertas ini merupakan kajian semula dan ulasan terhadap kajian yang pernah dilakukan sebelum ini tentang pengecaman teks Arab. Model yang dibina dibahagikan kepada lima tahap. Kaedah bagi setiap tahap dibincangkan mengikut implementasi oleh teknik yang menggunakan sampel teks Arab yang diterjemahkan daripada pelbagai sumber.

KAJIAN 3

Tajuk: Segmentation of Printed Arabic Text

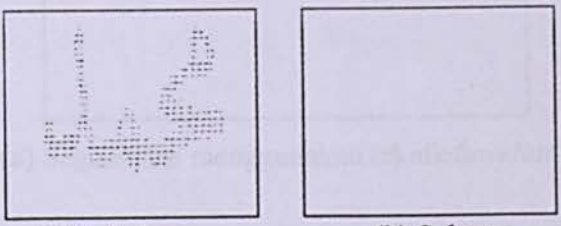
Kajian oleh: Adnan Amin

Methodologi yang digunakan:

Terdapat dua teknik yang telah diaplikasikan untuk segmentasi perkataan Arab yang ditulis dengan tangan mahupun percetakan mesin iaitu segmentasi secara tersirat (*implicit*) dan nyata (*explicit*). Dalam teknik segmentasi tersirat, perkataan akan disegmenkan terus kepada huruf. Segmentasi jenis ini biasanya direkabentuk bersama peraturan yang boleh mengenalpasti kesemua titik aksara yang disegmenkan.

Bagi teknik segmentasi nyata, perkataan disegmenkan secara luaran kepada *pseudo-letters* di mana seterusnya proses kenalpasti dilakukan secara individu. Teknik ini boleh dibahagikan kepada tiga langkah. Langkah pertama ialah pendigitalan dan prapemprosesan di mana imej asal ditukarkan kepada imej binari menggunakan pengimbas 300 *dots per inch* (dpi) dan penghasilan imej yang berkualiti.

Langkah kedua ialah mengesan skeleton bagi imej dari kanan ke kiri dan membentuk pepohon binari . *Thinning* adalah proses asas bagi operasi prapemprosesan dalam analisa imej yang ditakrifkan sebagai proses yang mengurangkan lebar garisan. Keputusan imej yang diperolehi dipanggil skeleton. **Rajah 2.3** menunjukkan contoh perkataan Arab sebelum dan selepas *thinning*.



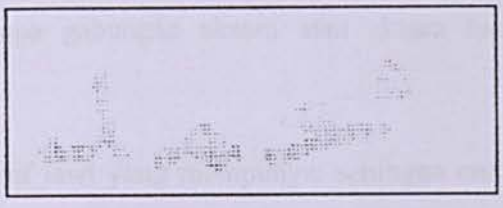
(a) Sebelum (b) Selepas

Rajah 2.3: Imej sebelum dan selepas proses *thinning* dilakukan.

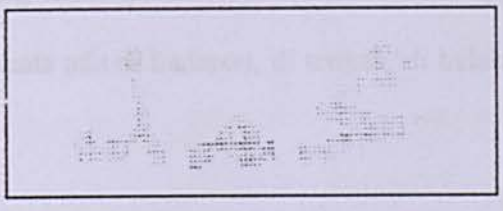
Apabila skeleton dapat dikesan, maka pepohon binari dibentuk termasuk ia mengandungi semua maklumat yang menerangkan stuktur imej tersebut. Struktur imej ditulis menggunakan primitif seperti garisan, gelung, dan dwigelung.

Akhir sekali, algoritma segmentasi digunakan untuk mengsegmenkan pepohon binari kepada subpepohon di mana setiap subpepohon menerangkan ciri-ciri imej. Fasa ini adalah perlu dan langkah genting dalam pengecaman tulisan Arab. **Rajah 2.4** menunjukkan beberapa teknik yang digunakan untuk membuat segmentasi.

(a) Segmentasi menggunakan teknik *baseline*



(b) Segmentasi yang dilakukan secara teliti



(c) Segmentasi secara nyata.

Rajah 2.4: Beberapa teknik yang digunakan untuk proses segmentasi

Kesimpulan:

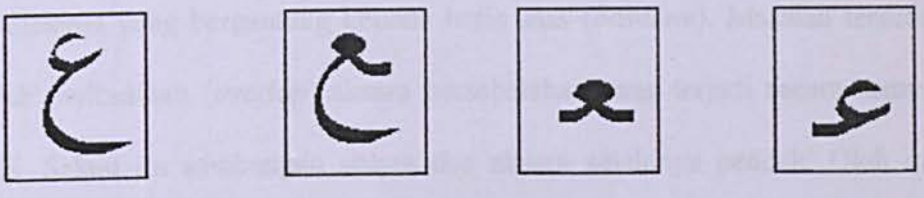
Teknik ini merupakan algoritma yang baru untuk pensegmenan perkataan Arab kepada huruf tunggal. Algoritma ini boleh diaplikasikan kepada mana-mana jenis tulisan dan ia membenarkan pertindihan aksara.

Proses *thinning* digunakan untuk menghasilkan perwakilan struktur yang sebenar bagi imej dengan mengesan skeleton. Walau bagaimapun, melihat kepada proses *erosion* terhadap imej, beberapa aksara tidak disegmenkan dengan teliti.

3. ANALISA SISTEM

Tulisan jawi ditulis dari kanan ke kiri. Ia merupakan satu tulisan yang unik. Dalam satu perkataan jawi, ia mempunyai aksara yang bersambung yang menyukarkan proses pengecaman. Terdapat ruang kosong yang mengasingkan beberapa gabungan aksara atau aksara tunggal bagi satu perkataan.

Terdapat juga huruf jawi yang mempunyai sehingga empat bentuk yang berlainan bagi mewakili aksara yang sama. Misalnya huruf a'in (ع)seperti dalam **Rajah 3.1** yang menunjukkan bentuk aksara jawi yang bergantung pada kedudukannya sama ada di hadapan, di tengah, di belakang atau secara tunggal.



a) Tunggal b) Belakang c) Tengah d) Hadapan

Rajah 3.1: Bentuk aksara jawi ع yang berlainan kedudukan.

Sesetengah aksara jawi mempunyai bentuk yang sama dengan aksara jawi yang lain. Ia dibezakan antara satu sama yang lain hanya dengan penambahan aksara tambahan misalnya titik atau huruf hamzah. Bilangan dan kedudukan titik dapat menentukan sesuatu huruf jawi tersebut. Sebagai contoh ialah huruf ba (ب) , ta (ت) dan tha (ث) seperti dalam **Rajah 3.2**.

(a) 1 titik (b) 2 titik (c) 3 titik

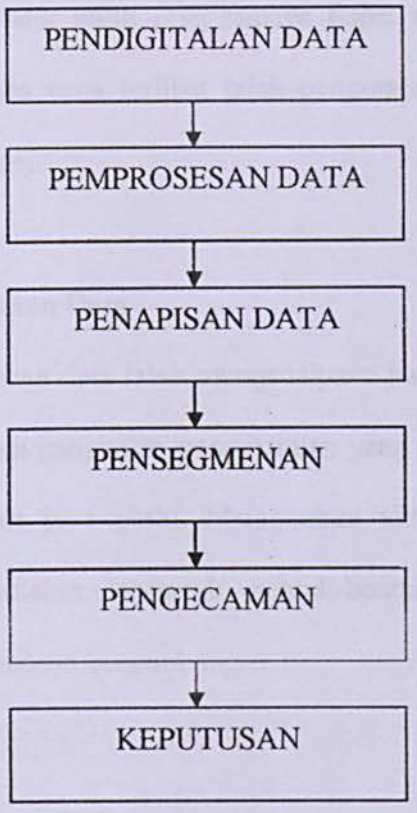
Rajah 3.2: Huruf yang mempunyai bentuk yang sama tetapi berbeza dari segi bilangan titik dan kedudukannya.

Penulisan jawi juga mempunyai saiz tulisan yang berlainan bagi jenis aksara jawi yang sama. Ia bergantung kepada gaya penulisan oleh individu-individu tertentu yang mempunyai kombinasi *vertical* yang dipanggil ligatur.

Terdapat dua masalah utama menggunakan teknik segmentasi tradisional yang bergantung kepada baris asas (*baseline*). Masalah tersebut ialah pertindihan (*overlap*) aksara bersebelahan yang terjadi secara semula jadi. Selain itu sambungan antara dua aksara selalunya pendek. Oleh itu, penempatan titik segmentasi adalah tugas yang sukar. Ciri ini menyukarkan penentuan sempadan bagi sesuatu huruf jawi.

3.1 KEPERLUAN FUNGSIAN

Keperluan fungsian terdiri daripada modul-modul yang diperlukan untuk membolehkan proses pengecaman tulisan jawi dilakukan. Ia melibatkan beberapa proses yang tertentu bermula daripada sesuatu imej diinput. Ia akan diikuti dengan pemprosesan terhadap imej tersebut dan seterusnya proses pensegmenan dan pengecaman akan dilaksanakan. **Rajah 3.3** menunjukkan proses-proses yang perlu dilaksanakan untuk tujuan pengecaman.



Rajah 3.3 : Proses pengecaman imej tulisan jawi

3.1.1 Proses Pendigitalan Data

Imej atau data akan diinput melalui proses imbasan menggunakan alat pengimbas. Ia bertujuan supaya imej dapat ditukar kepada bentuk digital. Imej tersebut akan disimpan ke dalam fail dalam format JPG, JPEG atau GIF.

3.1.2 Pemprosesan Data

Pemprosesan data bertujuan membuang data yang tidak diperlukan dan membaik pulih imej supaya dapat menghasilkan imej yang berkualiti. Proses yang terlibat ialah pengurangan gangguan dan peningkatan kualiti imej.

3.1.3 Proses Penapisan Data

Tujuan penapisan data ialah mengenalpasti jenis sesuatu aksara . Oleh sebab itu, sebuah pangkalan pengetahuan yang menyimpan semua bentuk tulisan tangan jawi perlu dibangunkan iaitu perwakilan kod rantaian. Aksara tersebut termasuk bentuk-bentuk yang berlainan sekiranya ia dalam keadaan bersambung.

3.1.4 Proses Pensegmenan

Pensegmenan memerlukan setiap aksara jawi dalam satu perkataan diasingkan kepada aksara yang bersambung dan tidak bersambung. Ini bermakna sekiranya terdapat ruang kosong, maka pengasingan aksara akan berlaku. Seterusnya bagi aksara yang bersambung, pemecahan bagi setiap aksara akan dilakukan.

3.1.5 Proses Pengecaman

Proses pengecaman adalah penting untuk mengenalpasti sesuatu aksara jawi. Ia dilaksanakan dengan membuat pemadanan kod rantaian aksara jawi yang diperolehi dengan kod rantaian yang disimpan dalam pangkalan pengetahuan. Proses pemadanan ini penting bagi menentusahkan proses pengecaman tulisan jawi.

3.1.6 Keputusan

Output yang akan terhasil daripada proses pengecaman ialah paparan aksara jawi tunggal yang mewakili perkataan jawi yang telah diinput. Ia menentukan sama ada keseluruhan proses pengecaman tulisan jawi berjaya dilaksanakan atau sebaliknya.

3.2 KEPERLUAN BUKAN FUNGSIAN

Keperluan bukan fungsian menyatakan kekangan atau had terhadap kajian yang dilakukan. Ia melibatkan imej yang akan diproses untuk tujuan pengecaman. Salah satu keperluan fungsian ialah saiz imej yang diperlukan adalah 512 x 512 bilangan piksel. Ia bertujuan supaya imej yang akan diperolehi lebih jelas.

Format fail imej yang digunakan ialah berbentuk *bitmap.* Ia diwakili oleh permodelan *I(r,c)* di mana *r* ialah baris dan *c* ialah lajur. Terdapat data piksel dan nilai kecerahan yang sama disimpan pada suatu format fail.

Perwakilan bagi imej yang diinput perlu ditukar dalam bentuk *gray-scale.* Pemprosesan imej dalam bentuk tahap kelabu adalah lebih sesuai digunakan. Ini kerana ia bergantung kepada teknik yang akan digunakan untuk pemprosesan imej.

3.3 KEPERLUAN PERKAKASAN DAN PERISIAN

3.3.1 Spesifikasi Perkakasan

Keperluan minima perkakasan yang diperlukan adalah seperti berikut:

- 255MHz Pentium II Processor
- 128.0MB RAM
- 512KB Internal L2 Cache
- 5GB Cakera Keras

3.3.2 Spesifikasi Perisian

Keperluan minima perisian yang diperlukan adalah seperti berikut:

- Microsoft Windows 98
- MATLAB 6.1 (Image Processing Tools)
- Microsoft Office 97

4. REKABENTUK SISTEM

4.1 TEKNIK YANG DIGUNAKAN

Terdapat beberapa teknik yang akan digunakan untuk membolehkan proses pengecaman tulisan jawi dilaksanakan. Teknik-teknik yang dipilih bergantung kepada proses tertentu yang akan dilalui oleh imej tersebut.

4.1.1 Teknik Pendigitalan Data

Imej akan diimbas dari kanan ke kiri menggunakan *scanner*. Ini kerana penulisan jawi bermula dari kanan ke kiri. Setelah imej disimpan ke dalam fail, ia secara automatik telah ditukar kepada imej berbentuk digital dan seterusnya pemprosesan imej akan dilakukan.

4.1.2 Teknik Pemprosesan Data

Alatan yang diperlukan dalam pemprosesan data ialah pengurangan ganggguan dan peningkatan kualiti imej. Teknik yang diperlukan untuk mengurangkan gangguan ialah penapis *Median*. Ia digunakan untuk mengurangkan gangguan jenis *salt and paper*.

Setiap piksel pada imej bergilir-gilir untuk melihat piksel kejiranannya untuk menentukan sama ada ia mewakili persekitaran imej atau tidak. Nilai median dikira bermula dengan menyisih semua nilai piksel kejiranannya kepada nombor yang disusun mengikut turutan menaik. Kemudian piksel yang dipertimbangkan digantikan dengan nilai piksel yang berada di tengah bagi susunan nombor tersebut.

Peningkatan kualiti imej memerlukan beberapa teknik untuk diaplikasikan. Ia terdiri daripada teknik untuk mengubah tahap kecerahan, melicinkan tepian, *threshold* dan *thinning*.

Perubahan tahap kecerahan terhadap imej akan menghilangkan kekaburan yang ada pada imej tersebut. Imej yang lebih jelas dan terang akan diperolehi.

Teknik melicinkan tepian bertujuan menghasilkan imej yang mempunyai tepian yang licin, rata dan kemas supaya keberkesanan proses seterusnya dapat dilihat.

Teknik *threshold* menukarkan imej tahap kelabu (*gray-scale*) dan imej berwarna (RGB) kepada imej perduaan (*binary*). Nilai *threshold* yang ditetapkan ialah 180. Apabila imej melalui proses *thresholding*, setiap piksel yang nilainya berada di atas daripada nilai *threshold* akan bertukar kepada warna putih yang mewakili nilai 1 dan piksel yang nilainya berada dibawah daripada nilai *threshold* akan bertukar kepada warna hitam iaitu 0.

Teknik *thinning* digunakan untuk menghasilkan imej yang nipis. Imej yang telah diproses dipanggil skeleton di mana ia mengurangkan imej asal kepada imej yang berbentuk garisan. Proses ini dinamakan *skeletonization*. Selain itu, satu proses yang bertindak mengecilkan sempadan imej untuk menjadikan ia lebih jelas dikenali sebagai *erosion*. Proses *skeletonization* dan *erosion* menggunakan penstrukturan elemen.

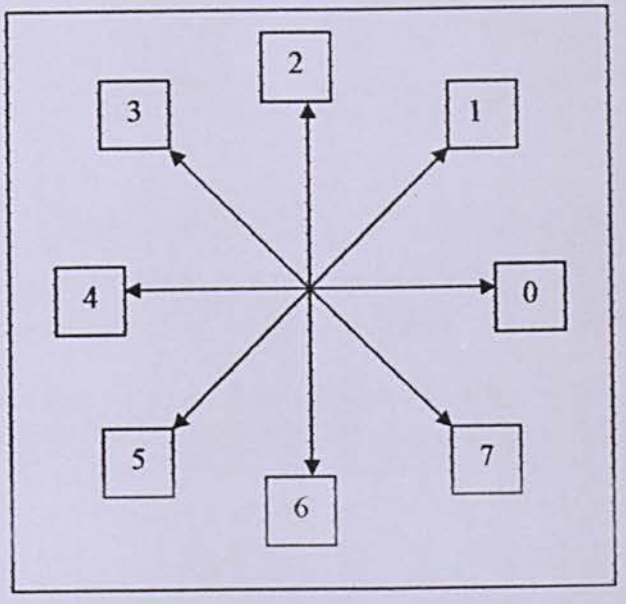
4.1.3 Teknik Pensegmenan

Imej mestilah dalam bentuk binari sebelum proses pensegmenan dilaksanakan. Teknik yang digunakan ialah *edge detection*. Fungsi ini dapat mengesan sempadan bagi sesuatu imej. Method Sobel dipilih untuk membolehkan sempadan dikenalpasti dan pengasingan aksara jawi yang bersambung dan tidak bersambung dapat dilakukan.

Pemecahan aksara-aksara jawi yang bersambung dilakukan dengan menggunakan teknik *Structural feature-based*.

4.1.4 Teknik Pengecaman

Pengecaman sesuatu huruf jawi menggunakan kaedah pembacaan kod rantaian atau *chain code*. Kod rantaian adalah jujukan nombor-nombor yang berdasarkan penakrifan gambarajah lapan arah seperti yang ditunjukkan pada **Rajah 4.1.** Teknik ini sesuai diaplikasikan kerana keadaan fizikal tulisan jawi mempunyai lengkuk, baris dan titik. Pembacaan kod rantaian hanya melibatkan aksara asas, sementara titik atau aksara tambahan seperti hamzah diabaikan.

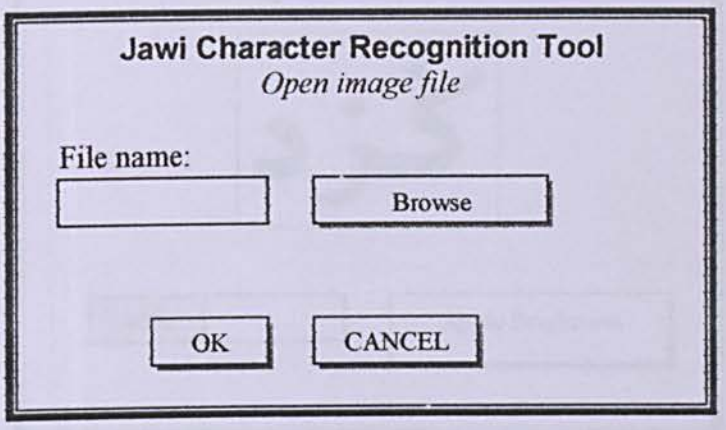


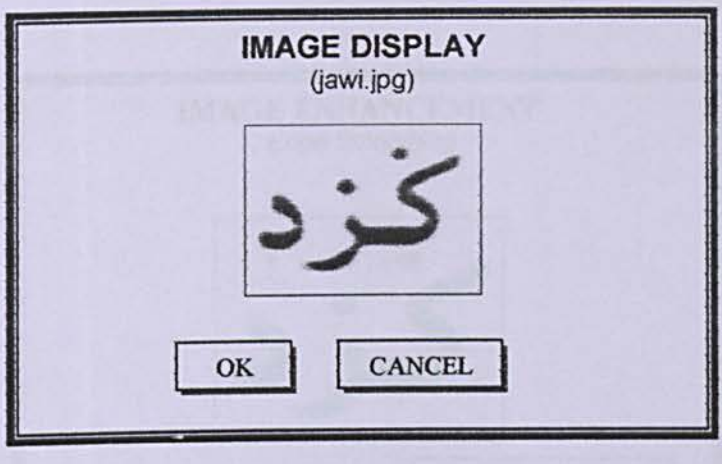
Rajah 4.1: Kaedah lapan arah kod rantaian.

Kaedah pengecaman bit dilaksanakan untuk mengesan aksara tambahan iaitu kewujudan titik atau hamzah. Kaedah ini juga dapat menentukan bilangan titik dan kedudukan titik tersebut sekiranya ada dengan mengenalpasti setiap koordinat yang diduduki oleh imej tersebut.

Setelah kod rantaian diperolehi hasil daripada pembacaan gambarajah arah bagi sesuatu aksara, proses pemadanan dilakukan untuk membandingkan kod rantaian yang disimpan dalam pangkalan pengetahuan. Proses pemadanan ini menggunakan teknik yang dipanggil *Dynamic Programming*.

4.2 REKABENTUK ANTARAMUKA





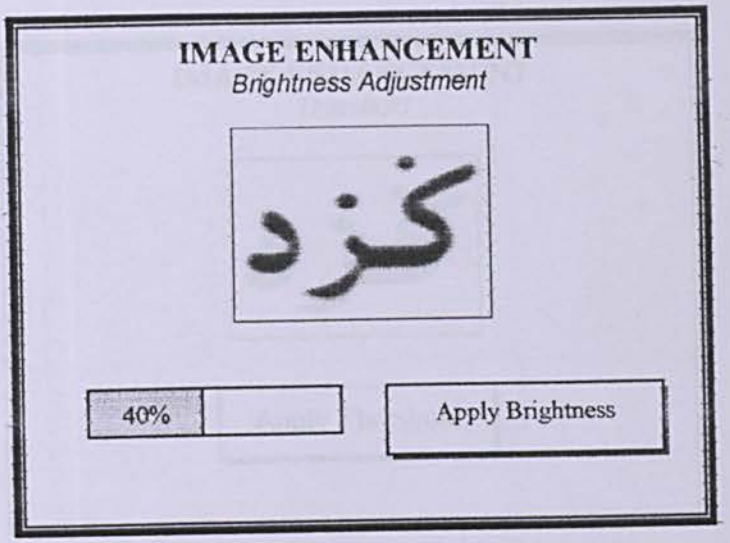


IMAGE ENHANCEMENT
Brightness Adjustment
40%
Apply Brightness

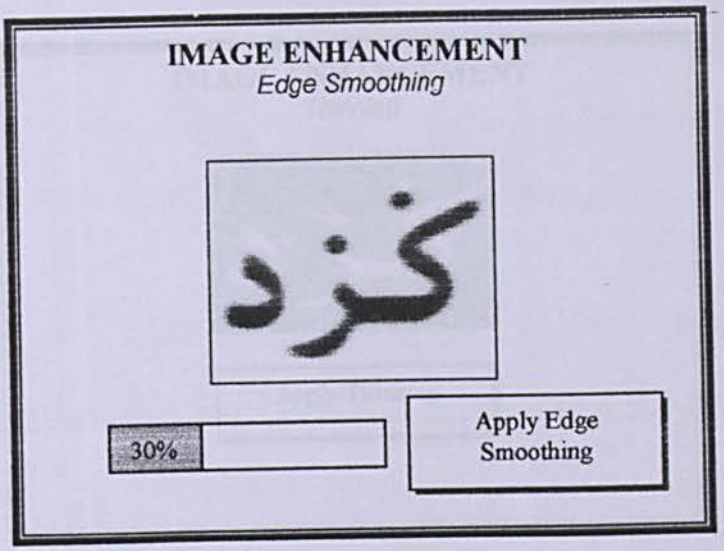


IMAGE ENHANCEMENT
Edge Smoothing
30%
Apply Edge Smoothing

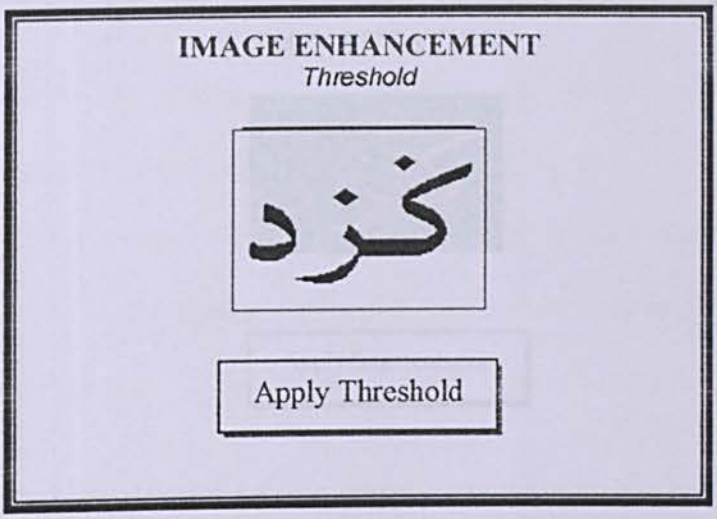


IMAGE ENHANCEMENT
Threshold
Apply Threshold

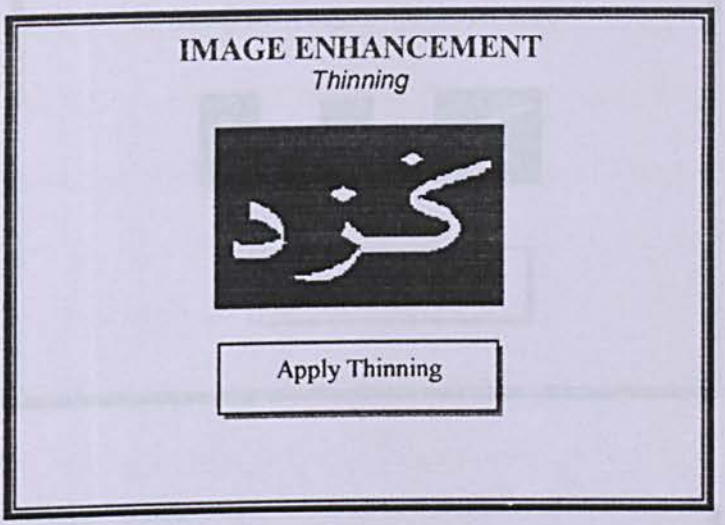


IMAGE ENHANCEMENT
Thinning
Apply Thinning

SEGMENTATION

Apply Segmentation

RECOGNITION

Result from segmentation process:

Apply Recognition

RESULT

ک ز د

Succesful ?

YES NO

5. ALGORITMA PENGECAMAN

Algoritma pengecaman imej terbahagi kepada modul pemprosesan imej, proses pensegmenan, dan pengecaman bagi satu huruf jawi.

Dalam pemprosesan imej, pada mulanya imej yang telah disimpan di dalam direktori akan dibuka dan dipaparkan. Pemprosesan terhadap imej melibatkan pengurangan gangguan jenis *salt and paper* menggunakan penapis *median*.

Pemprosesan imej juga adalah termasuk peningkatan kualiti imej yang menghasilkan imej yang berkualiti dari segi kecerahan (*brightness*). Seterusnya proses *threshold* dan *thinning* dilakukan terhadap imej tersebut.

Modul pensegmenan dilaksanakan terhadap satu huruf jawi bagi mengasingkan aksara asas dan aksara tambahan. Aksara asas menggambarkan struktur bagi huruf jawi manakala aksara tambahan mewakili titik yang menentukan identiti bagi huruf jawi tersebut.

Modul pengecaman terbahagi kepada dua bahagian iaitu pengecaman aksara asas dan aksara tambahan. Pengecaman aksara asas melibatkan penentuan titik mula dan pembacaan kod rantaian. Titik mula yang dikenal pasti iaitu dari kanan ke kiri adalah penting bagi membolehkan kod rantaian dapat dibaca. Selepas kod rantaian bagi aksara asas dapat dibaca, pemadatan terhadap kod rantaian tersebut dilakukan atau lebih dikenali sebagai *run length coding*.

Pengecaman aksara tambahan adalah bertujuan untuk menentukan bilangan titik yang dimiliki bagi sesuatu aksara jawi.

5.1 Pemprosesan Imej

5.1.1 Membuka fail dan memaparkan imej yang disimpan di dalam direktori.

```
[imagefile,imagepath]=uigetfile('*.*','Select your image');
[I,map]=imread([imagepath,imagefile]);
image(I);
```

5.1.2 Mengurangkan gangguan *salt and paper* menggunakan penapis *median*.

```
B=rgb2gray(I);
noise=medfilt2(B, [3 3]);
```

5.1.3 Mengubah tahap kecerahan imej

```
beta = ++.5; brighten(beta);
beta = .3; brighten(-beta);
```

5.1.4 Melaksanakan proses *threshold*

```
level=graythresh(noise);
threshold – im2bw(noise,level);
```

5.1.5 Melaksanakan proses *thinning*

```
SE = strel('square',2);
thinning – imdilate(threshold,SE)
```

5.2 Pensegmenan

```
[m,n] = size(thinning);
bw=I;
length_array–length(bw);
array = ones(1,length_array);
M = cell(length_array,1);
aksara_asas = cell(length_array,1);
aksara_tambahan=cell(length_array,1);

% -------- Masukkan imej (binary=0) ke dalam cell array M -------- %
for i=1:m
 total=0;
 for j=1:length_array
        if bw(i,j)--1   %Tentukan bilangan j–1
         total=total+1;
 end
 end
      if total ==length_array
        total=0;
         elseif total~=length_array
           for a=1:length_array
            array(1,a)=bw(i,a);
           end
           M{i} = array;
           total=0;
         end
 end
```

```
% ---------- Proses Segmentation ---------- %
temp=0;
 temp1=0;
 temp2=0;
 temp3=0;
 temp4=0;
  temp5=0;
 for i=1:m
see=isempty(M{i});
if see==1   %kalau M{i} tidak mempunyai nilai 0
        temp=temp+1;
         temp1=temp;

elseif see==0  %kalau M{i} mempunyai nilai 0
temp=temp+1;
temp2=temp;
break
end
end
temp3=temp2;
array2=ones(1,n);
aksara_asas{temp3-1}=array2;
        for i=temp3:m
         see=isempty(M{i});
        if see==0
        aksara_asas{i}=M{i};
         else
        aksara_asas{i}=array2;
        temp4=temp4+i;
```

```
            break
            end
        end
    temp5=temp4;
    for i=temp5:m
      aksara_tambahan{i}=M{i};
    end

    y = cell2mat(aksara_asas);
    figure,imshow(y),title('Aksara Asas');

    z=cell2mat(aksara_tambahan);
    figure,imshow(z);
```

5.3 Pengecaman Aksara

5.3.1 Pengecaman Aksara Asas

Pembacaan Kod Rantaian

```
[k,l]=size(y);

% ---------- Determine initial point for image ---------- %
length_array=length(l);
for i=1:k
   for j=l:-1:1
      if I(i,j)==0
      start1=i;
      start2=j;
       disp(l)
   fprintf(1,'Titik mula bagi imej adalah pada piksel (%.1d , %.1d )
         \n',start1,start2)
```

```
 break;
 end
 end
 if I(i,j)==0
  break;
 end
end
prev_code=8;
% ---------- Determine the chain code for image ---------- %

store  code=[];
increment=1;
while start1~=0 & start2~=0
if prev_code==8
if I(start1+1,start2-1)==0 %code=5
      start1=start1+1;
      start2=start2-1;
      store_code{increment}=5;
      prev_code=5;
   elseif I(start1+1,start2)==0 %code=6
       start1=start1+1;
      start2=start2;
 store_code{increment}=6;
 prev_code=6;
elseif I(start1-1,start2)==0 %code=2
 start1=start1-1;
 start2=start2;
store_code{increment}=2;
 prev_code=2;
 elseif I(start1,start2+1)==0 %code=0
```

```
start1=start1;
start2=start2+1;
 store_code{increment}=0;
prev_code=0;
elseif I(start1-1,start2+1)==0 %code=1
start1=start1-1;
start2=start2+1;
 store_code{increment}=1;
prev_code=1;
 elseif I(start1-1,start2-1)==0 %code=3
start1=start1-1;
start2=start2-1;
store_code{increment}=3;
prev_code=3;
 elseif I(start1,start2-1)==0 %code=4
 start1=start1;
start2=start2-1;
 store_code{increment}=4;
 prev_code=4;
elseif I(start1+1,start2+1)==0 %kod=7
 start1=start1+1;
start2=start2+1;
store_code{increment}=7;
 prev_code=7;
 else
  start1=0;
 start2=0;
 end
 increment=increment+1;
else % ********** if prev_code is exist - after defining the previous code
```

```
    if prev_code==5 % ********** if prev_code==5
      if I(start1+1,start2+1)==0 %kod=7 - prev_code=5
     start1=start1+1;
   start2=start2+1;
   store_code{increment}=7;
   prev_code=7;
elseif I(start1+1,start2)==0 %code=6 - prev_code=5
   start1=start1+1;
   start2=start2;
   store_code{increment}=6;
   prev_code=6;
elseif I(start1+1,start2-1)==0 %code=5 - prev_code=5
   start1=start1+1;
   start2=start2-1;
   store  code{increment}=5;
   prev_code=5;
elseif I(start1,start2-1)==0 %code=4 - prev_code=5
   start1=start1;
   start2=start2-1;
   store_code{increment}=4;
   prev_code=4;
elseif I(start1-1,start2-1)==0 %code=3 - prev_code=5
   start1=start1-1;
   start2=start2-1;
   store  code{increment}=3;
   prev_code=3;
elseif I(start1-1,start2)==0 %code=2 - prev_code=5
   start1=start1-1;
   start2=start2;
   store_code{increment}=2;
```

```
    prev_code=2;
  else
    start1=0;
      start2=0;
     end
      increment=increment+1;
     elseif prev_code==6 % ****************************** if prev_code==6

 if I(start1+1,start2+1)==0 %kod=7 - prev_code=6
    start1=start1+1;
    start2=start2+1;
    store_code{increment}=7;
    prev_code=7;
  elseif I(start1+1,start2)--0 %code=6 - prev_code=6
    start1=start1+1;
    start2=start2;
    store_code{increment}=6;
    prev_code=6;
  elseif I(start1+1,start2-1)--0 %code=5 - prev_code=6
    start1=start1+1;
    start2=start2-1;
    store_code{increment}=5;
    prev_code=5;
  elseif I(start1,start2-1)--0 %code=4 - prev_code=6
    start1=start1;
    start2=start2-1;
    store_code{increment}=4;
    prev_code=4;
  elseif I(start1-1,start2-1)==0 %code=3 - prev_code=6
     start1=start1-1;
```

```
    start2=start2-1;
    store_code{increment}=3;
    prev_code=3;
  else
    start1=0;
    start2=0;
  end
  increment=increment+1;
elseif prev_code==2 % ************************************ if prev_code==2
  if I(start1-1,start2-1)==0 %code=3 - prev_code=2
    start1=start1-1;
    start2=start2-1;
    store_code{increment}=3;
    prev_code=3;
  elseif I(start1-1,start2)==0 %code=2 - prev_code=2
    start1=start1-1;
    start2=start2;
    store_code{increment}=2;
    prev_code=2;
  elseif I(start1-1,start2+1)==0 %code=1 - prev_code=2
    start1=start1-1;
    start2=start2+1;
    store_code{increment}=1;
    prev_code=1;
  elseif I(start1,start2+1)==0 %code=0 - prev_code=2
    start1=start1;
    start2=start2+1;
    store_code{increment}=0;
    prev_code=0;
  elseif I(start1+1,start2+1)==0 %kod=7 - prev_code=2
```

```
        start1=start1+1;
        start2=start2+1;
        store_code{increment}=7;
        prev_code=7;
    else
        start1=0;
        start2=0;
    end
    increment=increment+1;
elseif prev_code==0 % ************************************ if prev_code==0
    if I(start1-1,start2+1)==0 %code-1 - prev_code-0
        start1=start1-1;
        start2=start2+1;
        store_code{increment}-1;
        prev_code=1;
    elseif I(start1,start2+1)==0 %code=0 - prev_code=0
        start1=start1;
        start2=start2+1;
        store_code{increment}-0;
        prev_code=0;
    elseif I(start1+1,start2+1)==0 %kod=7 - prev_code=0
        start1=start1+1;
        start2=start2+1;
        store_code{increment}-7;
        prev_code=7;
    elseif I(start1+1,start2)==0 %code=6 - prev_code=0
        start1=start1+1;
        start2=start2;
        store_code{increment}=6;
        prev_code=6;
```

```
elseif I(start1+1,start2-1)==0 %code=5 - prev_code=0
    start1=start1+1;
    start2=start2-1;
    store_code{increment}=5;
    prev_code=5;
  else
    start1=0;
    start2=0;
  end
  increment=increment+1;
elseif prev_code==1 % ************************************ if prev_code--1
  if I(start1-1,start2-1)==0 %code=3 - prev_code=1
    start1=start1-1;
    start2=start2-1;
    store_code{increment}=3;
    prev_code=3;
  elseif I(start1-1,start2)==0 %code=2 - prev_code=1
    start1=start1-1;
    start2=start2;
    store_code{increment}=2;
    prev_code=2;
  elseif I(start1-1,start2+1)==0 %code=1 - prev_code=1
    start1=start1-1;
    start2=start2+1;
    store_code{increment}=1;
    prev_code=1;
  elseif I(start1,start2+1)==0 %code=0 - prev_code=1
    start1=start1;
    start2=start2+1;
    store_code{increment}=0;
```

```
    prev_code=0;
  elseif I(start1+1,start2+1)==0 %kod=7 - prev_code=1
    start1=start1+1;
    start2=start2+1;
    store_code{increment}=7;
    prev_code=7;
  elseif I(start1+1,start2)==0 %code=6 - prev_code=1
    start1=start1+1;
    start2=start2;
    store_code{increment}=6;
    prev_code=6;
  else
    start1=0;
    start2=0;
  end
  increment=increment+1
elseif prev_code==3 % ********************************** if prev_code==3
  if I(start1+1,start2-1)==0 %code=5 - prev_code=3
    start1=start1+1;
    start2=start2-1;
    store_code{increment}=5;
    prev_code=5;
  elseif I(start1,start2-1)==0 %code=4 - prev_code=3
    start1=start1;
    start2=start2-1;
    store_code{increment}=4;
    prev_code=4;
  elseif I(start1-1,start2-1)==0 %code=3 - prev_code=3
    start1=start1-1;
    start2=start2-1;
```

```
        store_code{increment}=3;
        prev_code=3;
    elseif I(start1-1,start2)==0 %code=2 - prev_code=3
        start1=start1-1;
        start2=start2;
        store_code{increment}=2;
        prev_code=2;
    elseif I(start1-1,start2+1)==0 %code=1 - prev_code=3
        start1=start1-1;
        start2=start2+1;
        store_code{increment}=1;
        prev_code=1;
    elseif I(start1,start2+1)==0 %code=0 - prev_code=3
        start1=start1;
        start2=start2+1;
        store_code{increment}=0;
        prev_code=0;
    else
        start1=0;
        start2=0;
    end
    increment=increment+1;
elseif prev_code==4 % ************************************ if prev_code==4
    if I(start1+1,start2-1)==0 %code=5 - prev_code=4
        start1=start1+1;
        start2=start2-1;
        store_code{increment}=5;
        prev_code=5;
    elseif I(start1,start2-1)==0 %code=4 - prev_code=4
        start1=start1;
```

```
    start2=start2-1;
    store_code{increment}=4;
    prev_code=4;
  elseif I(start1-1,start2-1)==0 %code=3 - prev_code=4
    start1=start1-1;
    start2=start2-1;
    store_code{increment}=3;
    prev_code=3;
  elseif I(start1-1,start2)==0 %code=2 - prev_code=4
    start1=start1-1;
    start2=start2;
    store_code{increment}=2;
    prev_code=2;
  elseif I(start1-1,start2+1)==0 %code=1 - prev_code=4
    start1=start1-1;
    start2=start2+1;
    store_code{increment}=1;
    prev_code=1;
  else
    start1=0;
    start2=0;
  end
  increment=increment+1;
elseif prev_code==7 %********************************** if prev_code==7
  if I(start1-1,start2+1)==0 %code=1 - prev_code=7
    start1=start1-1;
    start2=start2+1;
    store_code{increment}=1;
    prev_code=1;
  elseif I(start1,start2+1)==0 %code=0 - prev_code=7
```

```
    start1=start1;
    start2=start2+1;
    store_code{increment}=0;
    prev_code=0;
elseif I(start1+1,start2+1)==0 %kod=7 - prev_code=7
    start1=start1+1;
    start2=start2+1;
    store_code{increment}=7;
    prev_code=7;
elseif I(start1+1,start2)==0 %code=6 - prev_code=7
    start1=start1+1;
    start2=start2;
    store_code{increment}=6;
    prev_code=6;
elseif I(start1+1,start2-1)==0 %code=5 - prev_code=7
    start1=start1+1;
    start2=start2-1;
    store_code{increment}=5;
    prev_code=5;
elseif I(start1,start2-1)==0 %code=4 - prev_code=7
    start1=start1;
    start2=start2-1;
    store_code{increment}=4;
    prev_code=4;
else
    start1=0;
    start2=0;
end
increment=increment+1;
```

```
end %end for prev_code=5

 end % end for prev_code=8

end % end for while loop

S = cell2mat(store_code);

disp('Kod rantaian yang yang dibaca adalah:')
disp(S)
```

Pemadatan kod rantaian atau *run length coding.*

```
[o,p]=size(S); %mengambil parameter S daripada bacaan kod rantaian
Z=cell(o,1);
tempi=0;
tempii=1;
  i=o;          %membezakan turutan kod rantaian yang dibaca
  for j=1:p-1
     tempi=j;      %tempi memegang nilai j
    a=S(1,tempi);
     b=S(1,tempi+1);
     if a<=b
       T=b-a;
     else
       T=a-b;
     end
     if (T==0)
       Z{tempii}=a;
    else
     tempii=tempii+1;
```

```
 Z{tempii}=b;
end
tempi=j;
  end
  disp('Run length coding ialah: ')
  z=cell2mat(Z);
  disp(z)
```

5.3.2 Pengecaman Aksara Tambahan

```
[m,n]=size(z);
for i=1:m
 for j=1:n
        if z(i,j)==0
         temp1=i;
         temp2=j;
        break
         end
 end
end

temp3=temp1;
 temp4=temp2+1;
 for j=temp4:n
         if z(temp3,j)==1
         figure,imshow(z),title('Satu Titik');

        temp5=j;
        break;
         end
```

```
end

temp6=temp5+1;
        for j=temp6:n
                if z(temp3,j)==0
                figure,imshow(z),title('Dua Titik')
                end
         end
```

6. PENGUJIAN SISTEM

Pengujian akan terbahagi kepada beberapa modul yang melibatkan pelaksanaan fungsi-fungsi yang tertentu. Input bagi imej yang akan diuji ialah huruf ba. Selain itu, terdapat penghantaran parameter bagi pengujian pada modul tertentu.

6.1 Pemprosesan Imej

Pemprosesan imej terdiri daripada beberapa fungsi yang digunakan untuk menghasilkan imej yang berkualiti. Fungsi-fungsi ini kebanyakkannya telah ditakrifkan dalam *Image Processing Toolbox* iaitu salah satu peralatan bagi perisian MATLAB. Fungsi-fungsi yang digunakan adalah bergantung kepada proses yang akan dilaksanakan seperti berkut:

[imagefile,imagepath]=uigetfile('*.*','Select your image')

Memulangkan nama dan laluan bagi fail yang dipilih di dalam kotak dialog. Selepas butang *Open* ditekan, *imagefile* mengandungi nama bagi fail yang telah dipilih dan *imagepath* mengandungi nama laluan yang dipilih. Jika butang *Cancel* ditekan atau jika ralat terjadi, *imagefile* dan *imagepath* diset kepada 0.

[I,map]=imread([imagepath,imagefile])

Membaca indeks imej pada *imagepath* kepada I dan *map* yang berkaitan dengannya kepada map. Nilai *map* adalah diskalakan kepada julat [0,1]. *imagepath* adalah string yang menspesifikkan format bagi fail. Jika *imread* tidak boleh mencari nama fail *imagepath* , ia akan lihat nama fail *imagepath.fmt*.

B=rgb2gray(I)

Menukarkan nilai sebenar bagi imej I kepada *grayscale intensity* bagi imej B.

noise=medfilt2(B, [3 3])

Melaksanakan penapisan median bagi matriks B dalam dua dimensi. Setiap piksel output mengandungi nilai median dalam kejiranan 3 x 3 di sekitar piksel yang sama pada imej input.

brighten(beta)

Menggantikan *colormap* semasa dengan *colormap* yang cerah atau gelap bagi warna sama. Fungsi *brighten(beta)*, digunakan bersama fungsi *brighten(-beta)*, di mana beta adalah kurang daripada 1 dan akan menyimpan *map* asal.

level=graythresh(I)

Mengira imej *threshold* dengan menukarkan imej tahap kelabu ke imej binari bersama fungsi *im2bw*. *level* adalah nilai intensiti dalam julat [0 1].

threshold = im2bw(noise,level)

Menukarkan imej intensiti *noise* kepada warna hitam dan putih.

SE = strel('square',2)

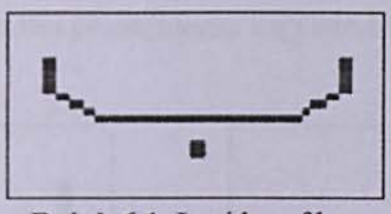
Untuk mencipta penstrukturan elemen (*structuring element*), SE bagi jenis bentuk (*shape*) yang telah dispesifikkan. Fungsi *strel* akan dilaksanakan berdasarkan parameter dan bentuk. Di dalam pengujian ini, bentuk *square* digunakan untuk membentuk penstrukturan elemen *square* di mana lebar bagi bentuk *square* ditentukan oleh parameter yang bernilai 2 yang mesti dalam skala integer bukan negatif.

thinning = imdilate(threshold,SE)

Menipiskan imej tahap kelabu, imej binari, atau imej *packed binary* bagi imej *threshold*, dan akan memulangkan imej nipis. Parameter SE adalah objek bagi penstrukturan elemen, atau tatasusunan bagi objek penstrukturan elemen yang dipulangkan oleh fungsi *strel*.

6.2 Pensegmenan

Proses pensegmenan melibatkan pemecahan aksara asas dan aksara tambahan. Pengujian proses pensegmenan melibatkan satu huruf jawi iaitu ba (ﺑ). Imej huruf ba yang akan diinput ke dalam modul pensegmenan ialah seperti dalam **Rajah 6.1**.



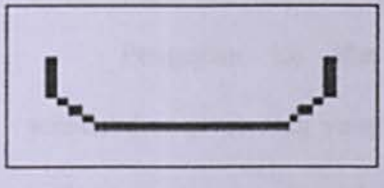
Rajah 6.1: Imej huruf ba.

Proses bermula apabila nilai bagi baris pertama di mana terdapat sekurang-kurangnya satu piksel yang bernilai sifar dipulangkan. Apabila nilai baris tersebut telah dikenalpasti, semua nilai piksel yang menduduki baris itu dan diikuti baris berikutnya akan disimpan ke dalam satu *cell* sehingga terdapat satu baris pertama yang mempunyai semua nilai piksel bersamaan dengan satu. Imej yang disimpan ke dalam *cell* ini adalah mewakili struktur aksara asas.

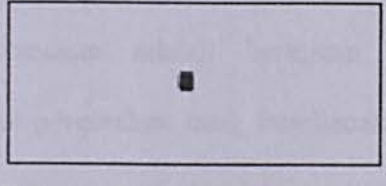
Baris yang mempunyai semua nilai piksel bersamaan dengan satu menujukkan terdapat ruang kosong di antaranya. Maka apabila nilai baris tersebut dipulangkan, piksel-piksel bagi baris tersebut dan baris seterusnya akan disimpan pada cell yang lain bagi mewakili struktur aksara tambahan.

Proses pensegmenan berjaya dilaksanakan kerana wujud dua cell yang berlainan yang menyimpan struktur aksara asas dan aksara tambahan. Seterusnya imej yang disimpan dalam dua *cell* yang berlainan akan melalui proses yang seterusnya iaitu proses pengecaman aksara.

Rajah di bawah menunjukkan dua *cell* yang mewakili struktur asas dan aksara tambahan selepas proses pensegmenan bagi huruf ba yang diuji.



Rajah 6.2: Struktur asas.



Rajah 6.3: Struktur tambahan.

6.3 Pengujian Pengecaman Aksara

Pengujian bagi pengecaman aksara terbahagi kepada dua bahagian iaitu pengecaman terhadap aksara asas dan aksara tambahan. Input bagi dua bahagian tersebut diperolehi daripada proses pensegmenan yang dilaksanakan sebelumnya.

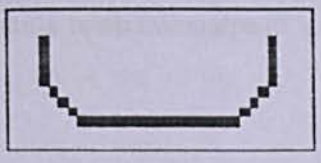
6.3.1 Pengecaman Aksara Asas

Pengujian Kod Rantaian

Pengujian ke atas kod rantaian adalah bertujuan untuk menentukan jenis arah yang mewakili pergerakan imej. Pembacaan bagi kod rantaian yang diperolehi merupakan perwakilan bagi stuktur asas huruf jawi tersebut.

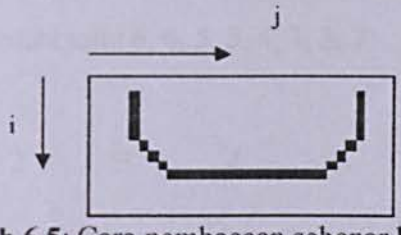
Kod rantaian akan dibaca mengikut arah jam. Berikut ditunjukkan kod rantaian dan bagaimana kod rantaian akan dibaca Oleh kerana kod rantaian ini dibaca hanya mengikut arah jam, maka akan wujud hanya lapan kemungkinan sahaja bagi setiap piksel dalam setiap satu kod rantaian.

Ujian kod rantaian hanya dilakukan ke atas aksara asas yang diperolehi daripada output proses pensegmenan iaitu seperti dalam **Rajah 6.4**.

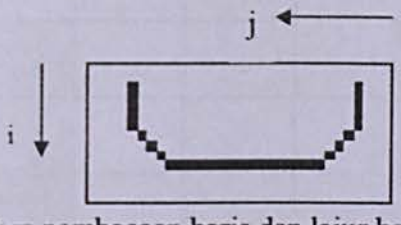


Rajah 6.4: Aksara asas daripada output modul pensegmenan.

Sebelum membaca kod rantaian, titik mula bagi imej dalam **Rajah 6.4** perlu ditentukan terlebih dahulu. Imej aksara jawi boleh dibaca dalam matriks *(i j)* di mana i mewakili baris dan j mewakili lajur bagi imej. Penentuan titik mula aksara asas mestilah dibaca dari arah kanan ke kiri seperti yang ditunjukkkan dalam **Rajah 6.6** yang berdasarkan bagaimana tulisan jawi ditulis.



Rajah 6.5: Cara pembacaan sebenar baris dan lajur.

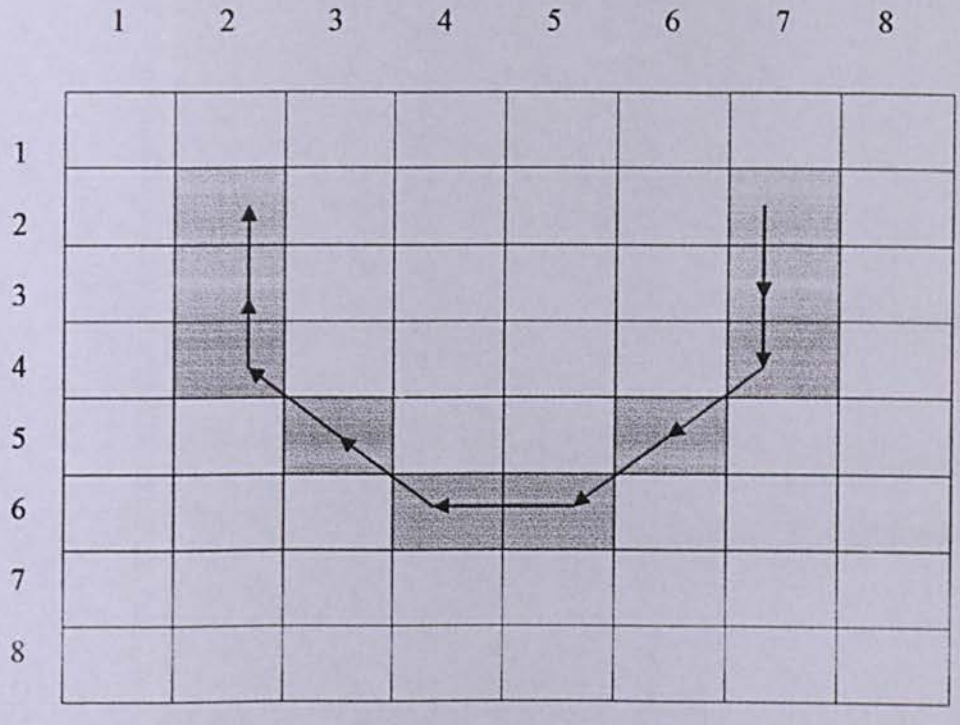


Rajah 6.6: Cara pembacaan baris dan lajur bagi penentuan titik mula.

Dalam algoritma titik mula, nilai pertama yang dijumpai dalam baris dan lajur *(i j)* akan memulangkan nilai titik mula bagi imej aksara asas tersebut. Algoritma akan berhenti daripada membaca nilai piksel yang seterusnya sebaik sahaja menjumpai nilai piksel pertama. Seterusnya, apabila titik mula telah dikenalpasti, pembacaan kod rantaian boleh dimulakan.

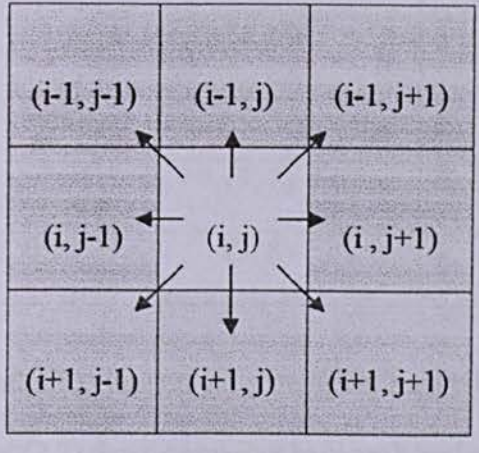
Seperti yang telah dinyatakan, pembacaan kod rantaian adalah berdasarkan kepada titik mula dan pergerakan titik seterusnya. Algoritma kod rantaian akan membaca titik mula dan memeriksa semua pergerakan yang mungkin yang mempunyai nilai pada piksel 8-kejiranan. Nilai titik mula tersebut akan berubah setiap kali pergerakan bagi titik yang lain dijumpai.

Rajah 6.7 di bawah digunakan bagi menunjukkan bagaimana kod rantaian diperolehi iaitu 6, 6, 5, 5, 4, 3, 3, 2.



Rajah 6.7: Piksel berwarna kelabu mewakili aksara asas ba yang bernilai 0 dan piksel berwarna putih bernilai 1. Anak panah merupakan bacaan kod rantaian.

Merujuk kepada **Rajah 6.7** , titik mula adalah pada nilai piksel *(2,7)*. Seterusnya berdasarkan kepada titik mula ini, algoritma akan mencari dengan memeriksa mana-mana piksel kejiranannya (8-kejiranan) yang mempunyai nilai 0 seperti dalam **Rajah 6.8**.



Rajah 6.8: Nilai piksel-piksel yang mungkin bagi piksel 8-kejiranan.

Seterusnya **Jadual 6.1** menunjukkan nilai-nilai piksel kejiranan yang mungkin bagi piksel titik mula yang bernilai (2,7) dan perwakilan bagi arah kod rantaiannnya.

Titik Mula *(i, j)*	Titik Seterusnya *(i, j)*	Arah kod rantaian
(2,7)	(2,8)	0 →
	(1, 8)	1 ↗
	(1,7)	2 ↑
	(1, 6)	3 ↖
	(2, 6)	4 ←
	(3, 6)	5 ↙
	(3, 7)	6 ↓
	(3, 8)	7 ↘

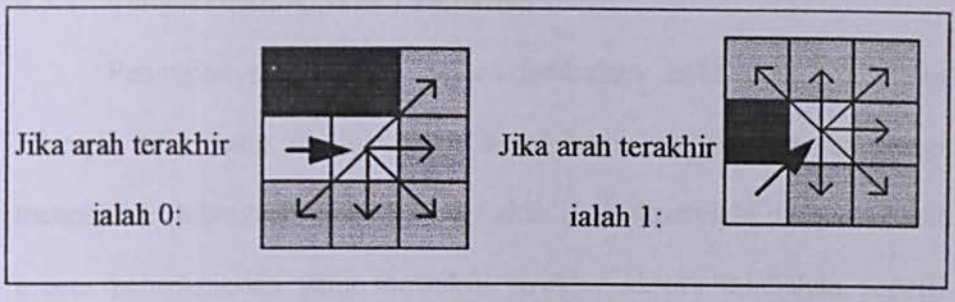
Jadual 6.1: Nilai-nilai piksel kejiranan yang mungkin bagi piksel titik mula yang bernilai (2,7) dan perwakilan bagi arah kod rantaiannnya.

Semasa pembacaan kod rantaian, adalah ditekankan di sini bahawa hanya arah kod rantaian tertentu sahaja akan dibaca dan ini adalah berdasarkan kepada titik mula. Ini disebabkan terdapat beberapa kod yang tidak mungkin dibaca untuk mewakili kod-kod rantaian. Ini dapat dijelaskan dengan merujuk pada **Jadual 6.2** dan **Rajah 6.9** di bawah.

Arah terakhir dibaca	Arah seterusnya (nilai ganjil)	Arah seterusnya (nilai genap)	Arah seterusnya (ikut arah jam)
0	1, 7, 0, 5, 6	0, 2, 1, 6, 7, 5	1, 0, 7, 6, 5
1	3, 1, 2, 7, 0, 6	2, 3, 0, 1, 6, 7	3, 2, 1, 0, 7, 6
2	3, 1, 2, 7, 0	2, 4, 3, 0, 1, 7	3, 2, 1, 0, 7
3	5, 3, 4, 1, 2, 0	4, 5, 2, 3, 0, 1	5, 4, 3, 2, 1, 0
4	5, 3, 4, 1, 2	4, 6, 5, 2, 3, 1	5, 4, 3, 2, 1
5	7, 5, 6, 3, 4, 2	6, 7, 4, 5, 2, 3	7, 6, 5, 4, 3, 2
6	7, 5, 6, 3, 4	6, 0, 7, 4, 5, 3	7, 6, 5, 4, 3
7	1, 7, 0, 5, 6, 4	0, 1, 6, 7, 4, 5	1, 0, 7, 6, 5, 4

Jadual 6.2: Arah seterusnya yang mungkin berdasarkan kepada arah terakhir dibaca.

Rajah 6.9 menunjukkan piksel yang berwarna hitam adalah piksel yang tidak mungkin mengandungi imej. Arah anak panah menunjukkan arah-arah seterusnya yang mungkin mengikut bacaan arah jam seperti yang telah ditunjukkan dalam **Jadual 6.1**.



Rajah 6.9: Piksel-piksel yang tidak mungkin iaitu piksel hitam dan arah kod rantaian yang mungkin diperiksa.

Pemadatan kod rantaian atau run length coding

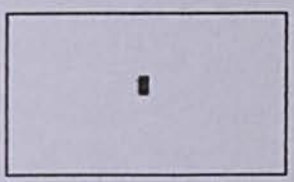
Algoritma pemadatan kod rantaian dibina untuk memadatkan nilai kod rantaian yang sama selepas pembacaan kod rantaian. Ini bertujuan bagi mengelakkan limpahan kod rantaian yang disimpan.

Satu *cell* akan diwujudkan bagi menyimpan kod rantaian yang akan diuji. Pengujian dilakukan dengan membezakan kod rantaian yang berturutan atau bersebelahan dengan operasi penolakan. Sekiranya baki yang diperolehi adalah selain daripada 0, kod rantaian tersebut akan disimpan ke dalam *cell*.

Bagi kod rantaian yang telah dibaca sebelum ini iaitu 6, 6, 5, 5, 4, 3, 3, 2 , maka *cell* yang telah diwujudkan akan menyimpan nilai 6, 5, 4, 3, 2.

6.3.2 Pengecaman Aksara Tambahan

Pengujian pengecaman aksara tambahan melibatkan penentuan bilangan titik yang dimiliki oleh huruf jawi tunggal iaitu ba. Bagi menentukan bilangan titik, imej yang akan diuji diperolehi daripada hasil proses pensegmenan yang mewakili struktur aksara tambahan seperti dalam **Rajah 6.10** di bawah.



Rajah 6.10: Aksara tambahan daripada output modul pensegmenan.

Perjalanan algoritma yang digunakan ialah dengan mengenal pasti satu baris pertama yang mempunyai nilai piksel 0 yang mewakili imej. Kemudian baris tersebut akan diperiksa sama ada terdapat nilai 0 berikutnya yang diasingkan oleh sekurang-kurangnya satu piksel bernilai 1.

Rajah 6.11 di bawah menunjukkan struktur aksara tambahan bagi huruf ba di mana baris yang akan disemak ialah baris yang ketiga yang mempunyai piksel bernilai 0 buat pertama kalinya dijumpai. Disebabkan tiada nilai 0 berikutnya yang diasingkan oleh sekurang-kurangnya satu ruang kosong, maka ini membuktikan bahawa ia mempunyai hanya satu titik.

Baris ke-3 →

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	0	0	1	1	1
1	1	1	0	0	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

Rajah 6.11: Struktur Aksara Tambahan ba yang diwakili oleh nilai piksel.

7. PERBINCANGAN

Keputusan yang diperolehi ialah alatan yang boleh digunakan untuk membantu proses pengecaman aksara jawi. Alatan yang dibangunkan terdiri daripada pemprosesan imej, pensegmenan dan pengecaman bagi satu huruf jawi tunggal.

Terdapat kelebihan dan kelemahan yang telah dikenalpasti dalam menghasilkan alatan pengecaman aksara jawi ini. Sekiranya dibandingkan dengan projek yang dilakukan sebelumnya, terdapat banyak kelebihan disebabkan penambahan ciri-ciri yang baru dan idea pelaksanaan yang lebih cekap seperti berikut:

- Pembinaan alatan yang khusus untuk pemprosesan imej iaitu pengurangan gangguan, dan peningkatan kualiti imej yang terdiri daripada pengubahsuaian tahap kecerahan, proses *threshold* dan *thinning*.
- Pembinaan alatan untuk tujuan pensegmenan adalah tanpa dipengaruhi oleh kedudukan aksara tersebut sama ada ia terletak di bahagian tengah, atas atau bawah. Pensegmenan berjaya dilaksanakan kerana ia dilakukan bukan secara manual dalam pengasingan aksara asas dan aksara tambahan.
- Pembinaan alatan untuk tujuan pengecaman adalah melibatkan aksara asas dan aksara tambahan. Penemuan titik mula yang diperolehi adalah amat penting bagi mengimplementasikan pembacaan kod rantaian. Algoritma kod rantaian yang dibina dapat dilaksanakan dengan jayanya dan ia adalah paling penting dalam menentukan proses pengecaman aksara jawi.

- Pembinaan alatan untuk pemadatan kod rantaian atau dikenali sebagai *run length coding* amat membantu meningkatkan keberkesanan dalam menyimpan kod rantaian yang dibaca. Ia adalah idea yang baru dan dapat diimplementasikan dalam proses pengecaman aksara jawi.
- Pembinaan alatan untuk pengecaman aksara tambahan tidak dipengaruhi oleh koordinat bagi kedudukan titik seperti yang telah diimplementasikan oleh projek sebelum ini. Alatan yang dibina tidak mensyaratkan bahawa satu titik mewakili satu koordinat atau satu piksel.

Kelemahan utama dalam pembinaan alatan pengecaman akasara jawi ini ialah tidak penghantaran parameter secara keseluruhannya iaitu dalam modul pemprosesan imej. Ini menyebabkan proses *thinning* tidak dapat dilaksanakan menggunakan imej bagi proses sebelumnya.

Selain itu, fungsi pengurangan gangguan dan pengubahsuaian tahap kecerahan yang dibina adalah hanya untuk pembuktian perjalanan fungsi tersebut. Imej yang terhasil tidak dihantar kepada proses yang seterusnya.

Dalam proses pengecaman aksara asas, imej yang diinput mestilah terdiri daripada satu piksel yang mewakili satu garis lurus. Kod rantaian tidak dapat dibaca sekiranya syarat ini tidak dipenuhi.

Pengujian yang dilakukan adalah dengan menggunakan satu aksara jawi tunggal iaitu huruf ب . Ini tidak menepati skop yang telah ditetapkan kerana tujuan utama yang ingin dicapai ialah menguji perlaksanaan algoritma yang dibangunkan.

Selain itu, rekabentuk antaramuka yang telah dibangunkan berlainan dengan apa yang telah dirancangkan. Ini kerana ia dapat memudahkan setiap perubahan imej dapat dilihat dengan lebih jelas apabila melalui proses seterusnya.

Peningkatan yang boleh dijalankan pada masa hadapan ialah dengan memastikan proses *thinning* dapat menghasilkan imej yang diperlukan dengan menggunakan parameter yang dihantar oleh proses sebelumnya.

Selain itu, bagi alatan pengecaman aksara asas melalui pembacaan kod ranatain, satu algoritma *fault tolerance* digunakan sekiranya imej yang diinput tidak terdiri daripada satu piksel yang diwakili oleh satu garis lurus. Kod rantaian yang dibaca akan diteliti dan dipilih bagi menentukan ia mewakili satu aksara jawi yang sebenar.

Cadangan saya ialah supaya projek ini dapat diteruskan dengan memperbaiki kelemahan yang sedia ada dan kajian yang lebih mendalam dapat dilaksanakan bagi menghasilkan alatan yang lebih cekap dan dapat direalisasikan ke atas perkasasan pencaman aksara jawi.

Kesimpulannya terdapat peningkatan alatan yang dihasilkan dan lebih berkesan berbanding projek sebelumnya. Pembangunan algoritma yang telah dilaksanakan amat penting bagi merealisasikan proses pengecaman aksara jawi khususnya bagi alatan pemprosesan imej iaitu *thinning*, pensegmenan dan pengecaman. Semoga kajian dan perlaksanaan projek ini dapat dimanfaatkan dalam membantu usaha pengecaman tulisan tangan jawi.

RUJUKAN

Adnan Amin. (2002). Structural Description to Recognising Arabic Characters Using Decision Tree Learning Techniques. *Springer-Verlag Berlin Heidelberg.***2396.**152-158.

Adnan Amin. (2001). Segmentation of Printed Arabic Text. *Springer-Verlag Berlin Heidelberg.***2013.**115-126.

Adnan Amin. (1998). Off-Line Arabic Character Recognition: The State of the Art. *Pattern Recognition.* **31(5).** 517-530.

M. S. Khorsheed. (2002). Off-Line Arabic Character Recognition –A Review. *Pattern Analysis & Applications.***5.**31-45

Omar Abdul Rahim. (2001). *Jawi Character Recognition.* Bsc. Thesis. University of Malaya.

Scott E Umbaugh.(1998). *Computer Vision and Image Processing.*Prentice Hall.

http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml

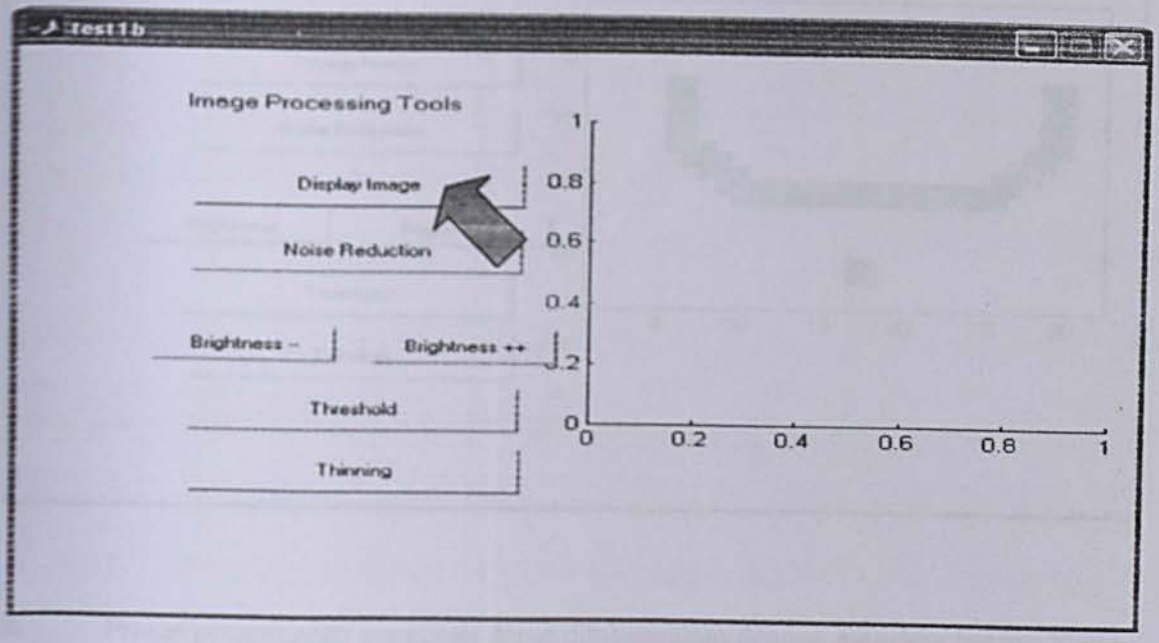
http://www.dai.ed.ac.uk/HIPR2/hipr_top.htm

http://www.dai.ed.ac.uk/CVonline/transf.htm

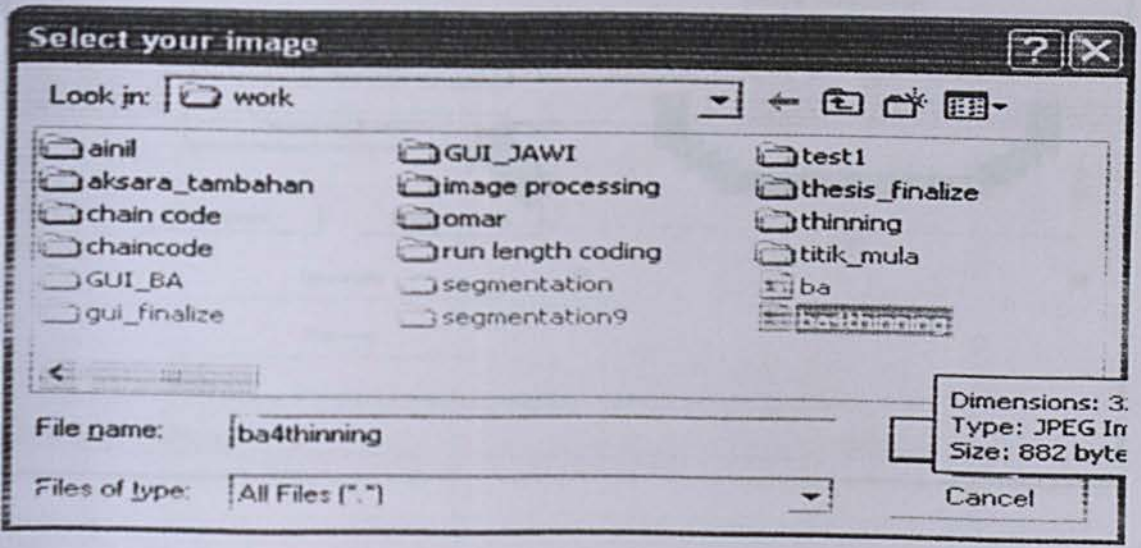
LAMPIRAN

MODUL PEMPROSESAN IMEJ

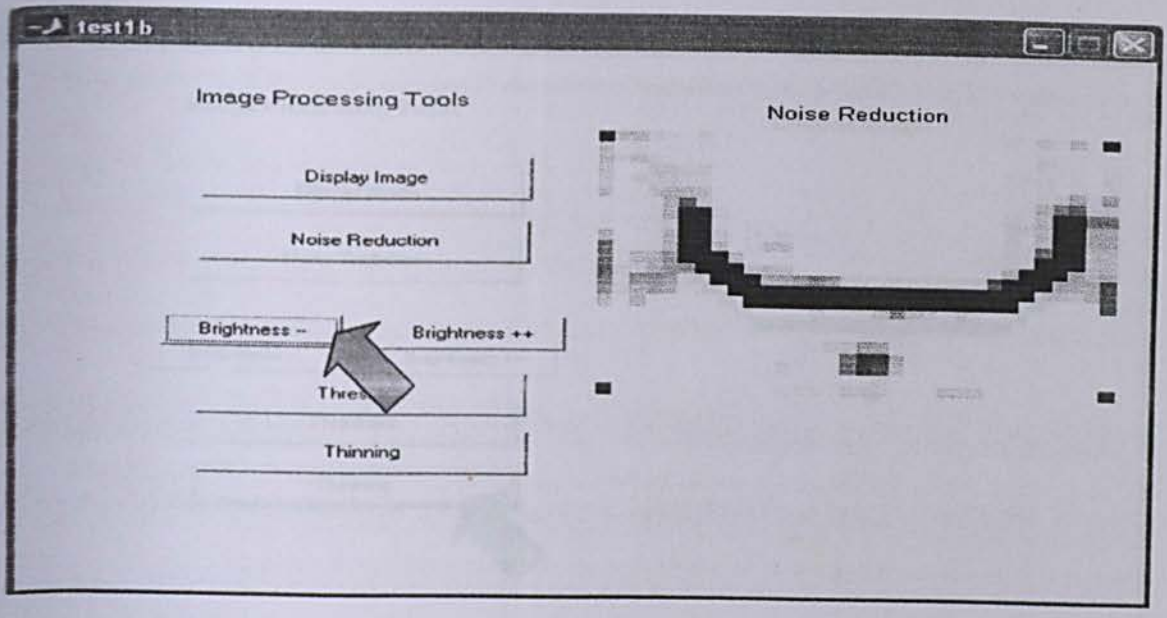
1. Bagi menginput imej butang *Display Image* diklik.



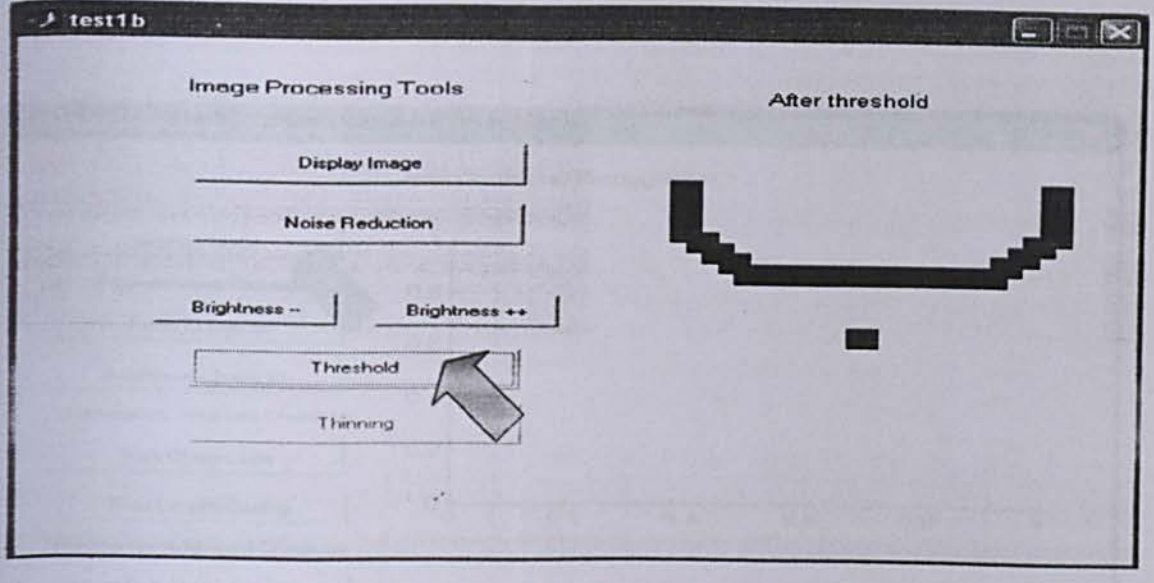
2. Pilih nama fail bagi imej yang akan diproses dari direktori dan tekan *Open.*



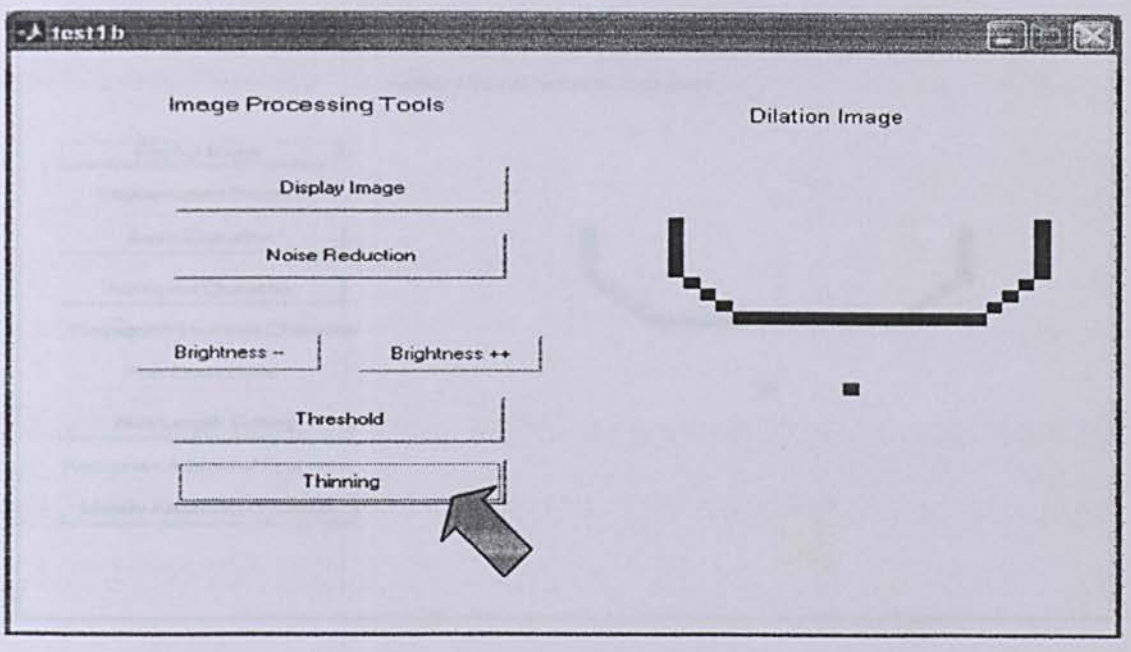
5. Klik ikon *Brightness* -- atau *Brightness* ++ untuk mengurangkan tahap kecerahan.



6. Klik ikon *Threshold* untuk menukarkan imej kepada nilai perduaan.

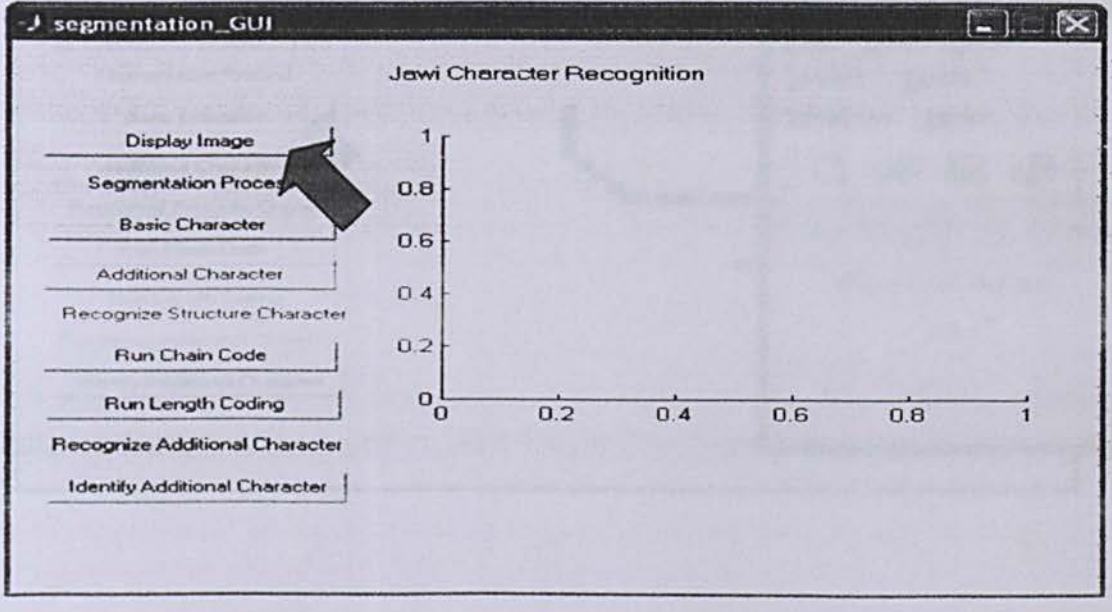


7. Seterusnya klik butang *Thinning* untuk menghasilkan imej garis lurus.

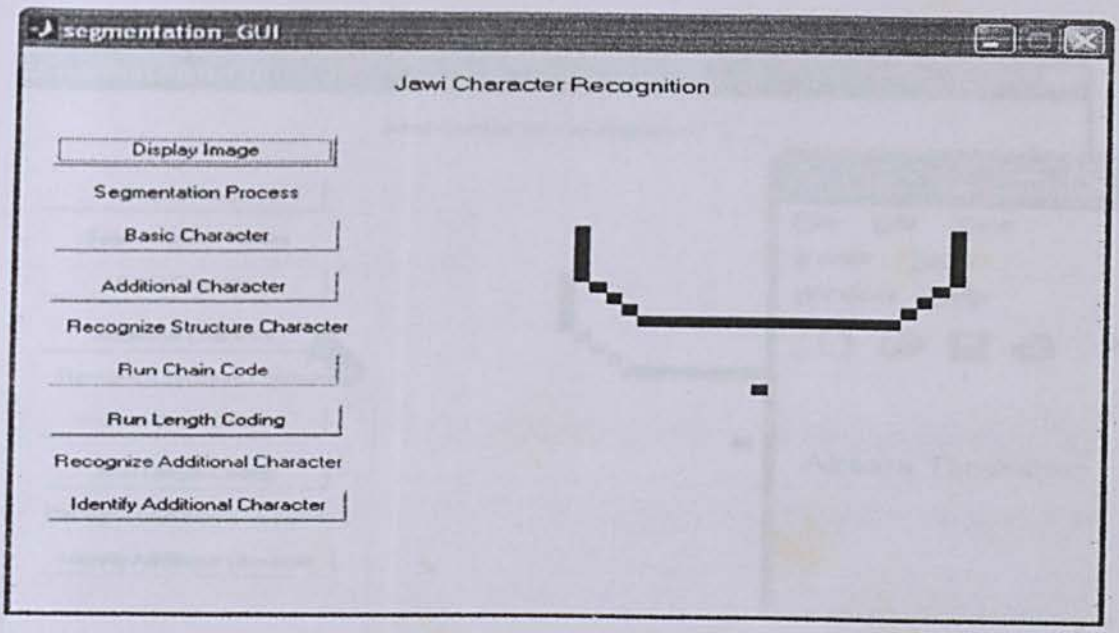


MODUL PENSEGMENAN DAN PENGECAMAN IMEJ

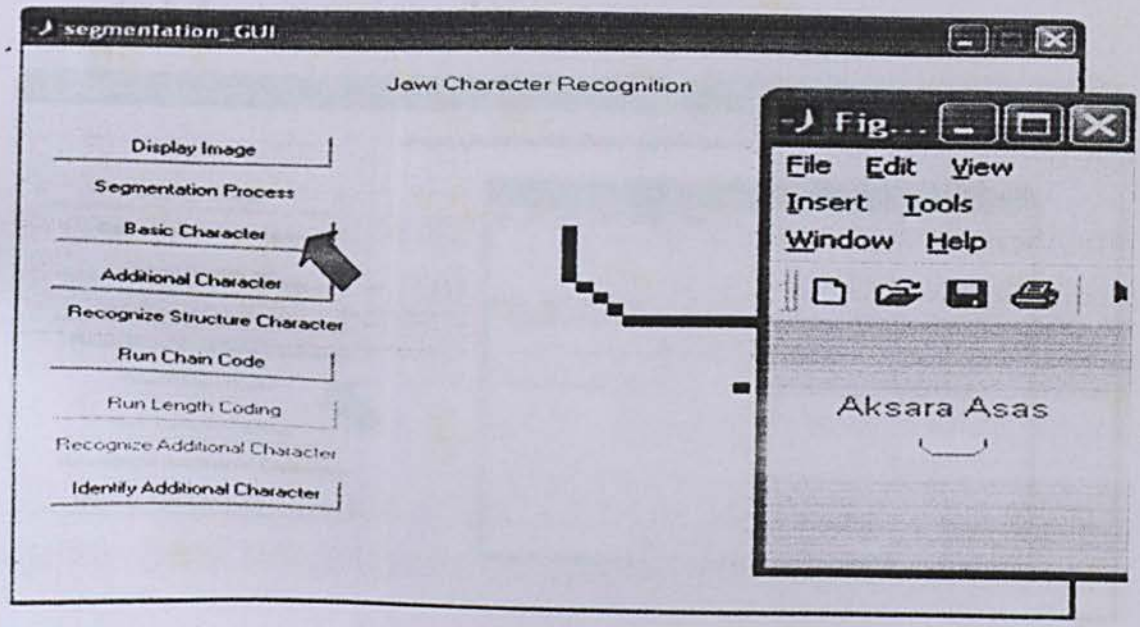
1. Klik butang *Display Image* untuk memaparkan input yang akan diproses.



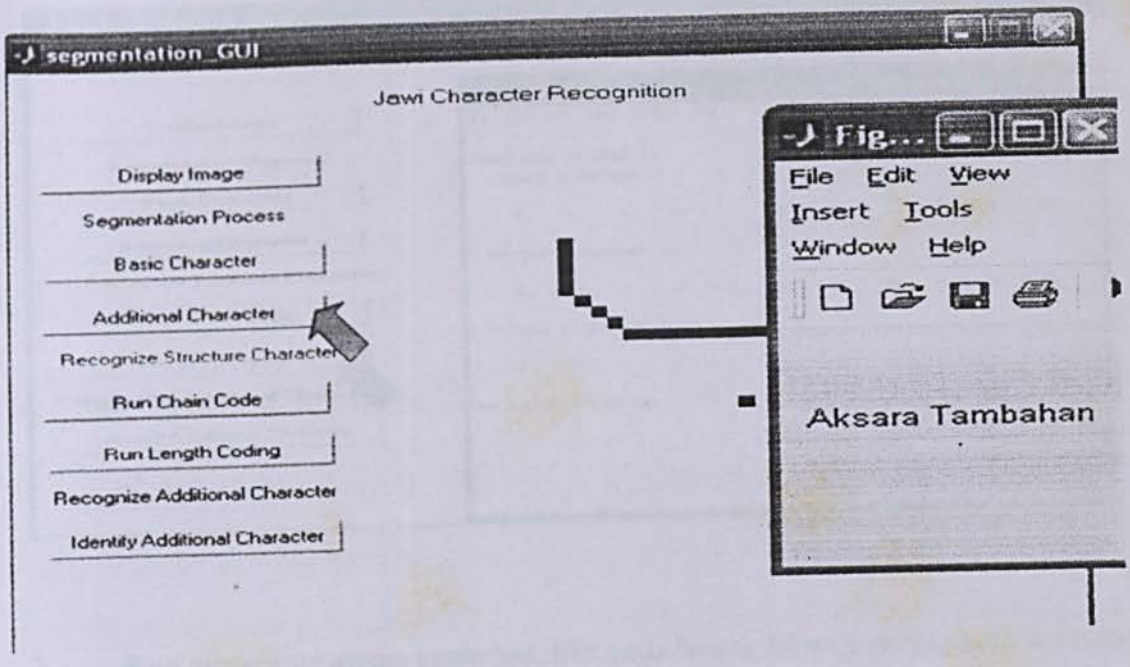
2. Skrin menunjukkan imej yang akan diproses.



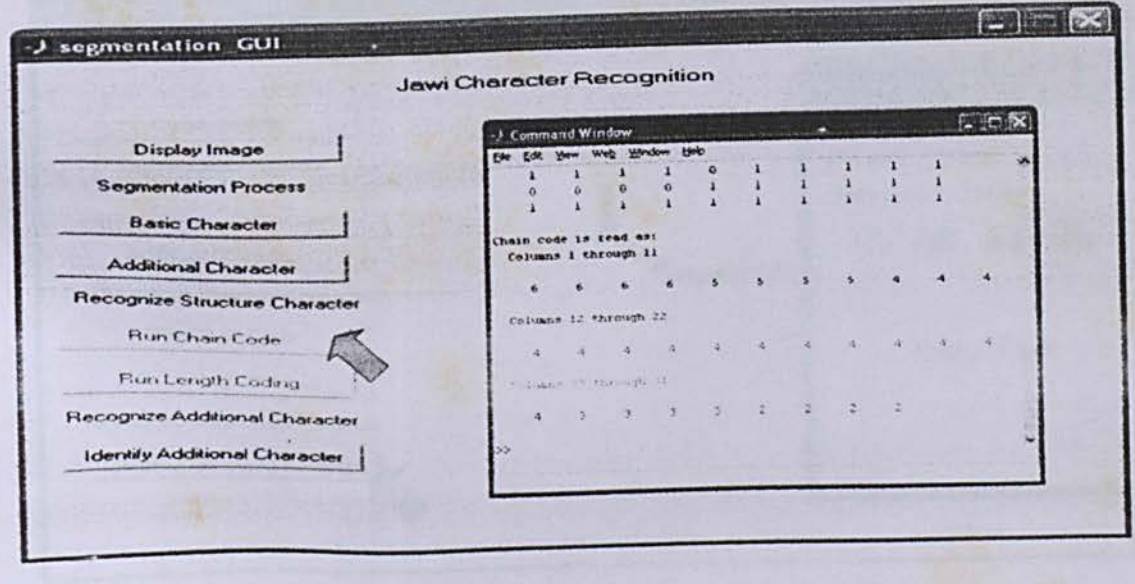
3. Pensegmenan bagi aksara asas dapat dilihat dengan menekan butang *Basic Character*.



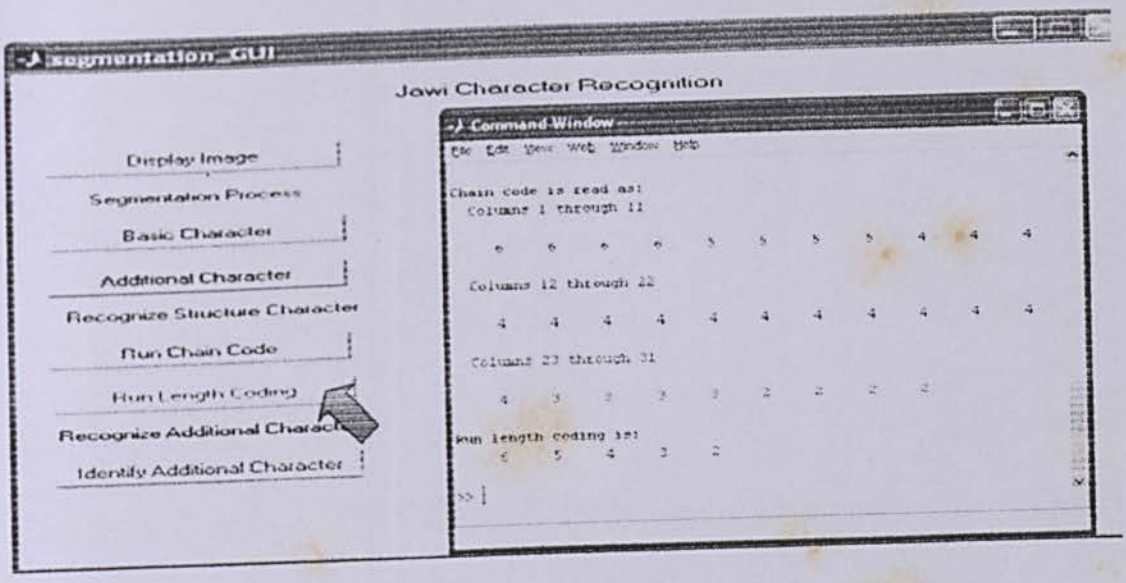
4. Pensegmenan bagi aksara tambahan dapat dilihat dengan menekan butang Additional *Character*.



5. Pembacaan kod rantaian dapat dilarikan dengan menekan ikon *Run Chain Code* dan kod rantaian yang diperolehi dapat dipaparkan.



6. Klik butang *Run Length Coding* bagi memadatkan pembacaan kod rantaian dan keputusan akan diperolehi.



7. Bagi mengecam aksara tambahan, klik pada butang *Identify Additional Character.* Keputusan kemudian akan dipaparkan pada skrin.